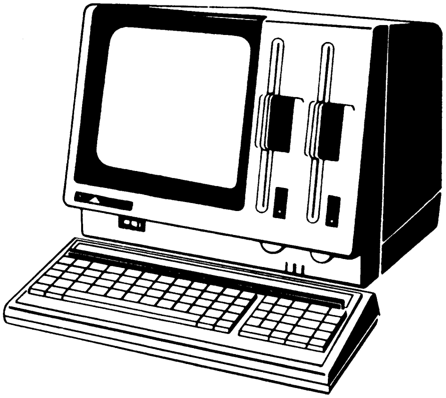
Advanced
Personal Computer
TM



MS-DOS System User's Guide

NEC
NEC Information Systems, Inc.
819-000101-6001 Rev. 00
6-83

Important Notice

(1) All rights reserved. This manual is protected by copyright. No part of this manual may be reproduced in any form whatsoever without the written permission of the copyright owner.
(2) The policy of NEC being that of continuous product improvement, the contents of this manual are subject to change, from time to time, without notice.
(3) All efforts have been made to ensure that the contents of this manual are correct; however, should any errors be detected, NEC would greatly appreciate being informed.
(4) NEC can assume no responsibility for errors in this manual or their consequences.

© Copyright 1983 by NEC Corporation

Contents

Page

Preface ix

Chapter 1 Introduction to MS-DOS

WHAT IS AN OPERATING SYSTEM? 1-1
WHAT IS MS-DOS? 1-1
  The MS-DOS Programs 1-2
  How MS-DOS Manages System Resources 1-2
    MEMORY 1-4
    DISKETTE DRIVES 1-5
    HARD DISK DRIVES 1-5
    THE PRINTER 1-5
    THE APC SCREEN 1-6
    AUXILIARY DEVICES 1-6
SOFTWARE PROVIDED ON YOUR MS-DOS DISKETTE 1-7

Chapter 2 Getting MS-DOS Started

THE APC STATUS LINE 2-1
MS-DOS VERSION NUMBER AND TIME AND DATE PROMPTS ... 2-2
THE MS-DOS SYSTEM PROMPT 2-3
ENTERING COMMANDS TO MS-DOS 2-4
  Command Types 2-4
  MS-DOS Command Format 2-4
  The Command Line 2-5
THE APC KEYBOARD 2-6
DRIVE DESIGNATIONS AND THE DEFAULT DRIVE 2-9
CHANGING THE DEFAULT DRIVE DESIGNATION 2-10
SINGLE-DRIVE SYSTEMS 2-10
MAKING A WORKING COPY OF THE MS-DOS DISKETTE 2-11
  Formatting a Diskette 2-12
  Copying From Diskette to Diskette 2-14
AUTOMATIC PROGRAM EXECUTION 2-15
HOW TO TURN THE SYSTEM OFF 2-15

Contents (cont'd)

Page

Chapter 3 MS-DOS Files

WHAT IS A FILE? 3-1
HOW ARE FILES CREATED? 3-1
FORMATTING DISKETTES AND HARD DISK BEFORE CREATING FILES 3-2
HOW MS-DOS KEEPS TRACK OF YOUR FILES 3-2
  The DIR (Show Directory) Command 3-3
  The CHKDSK (Check Disk) Command 3-4
HOW TO NAME YOUR FILES 3-4
WILD CARDS 3-7
  The ? Wild Card 3-7
  The * Wild Card 3-7
HOW TO COPY YOUR FILES 3-8
HOW TO PROTECT YOUR FILES 3-9
DIRECTORIES 3-9
FILENAMES AND PATHS 3-11
  Pathnames 3-12
  Pathing and External Commands 3-13
  Pathing and Internal Commands 3-13
DISPLAYING YOUR WORKING DIRECTORY 3-14
CREATING A DIRECTORY 3-15
HOW TO CHANGE YOUR WORKING DIRECTORY 3-16
HOW TO REMOVE A DIRECTORY 3-16

Chapter 4 MS-DOS Commands

TYPES OF MS-DOS COMMANDS 4-1
MS-DOS COMMAND FORMAT 4-2
INFORMATION COMMON TO ALL MS-DOS COMMANDS 4-3
THE TEMPLATE 4-4
CONTROL CHARACTER FUNCTIONS 4-7
BATCH PROCESSING 4-9
THE AUTOEXEC.BAT FILE 4-10
  How To Create An AUTOEXEC.BAT File 4-12
  Creating a .BAT File with Replaceable Parameters 4-12
  Executing a .BAT File 4-14

Release Bul. Addendum

Contents (cont'd)

country.com
crtdump
crttype
getsys
Hdbackup
Hdlist
Hdformat
[illegible]
[illegible]
ramdisk
Setcom
[illegible]

Page

HOW MS-DOS CONTROLS INPUT AND OUTPUT 4-15
  Redirecting Output 4-15
  Filters 4-16
  Command Piping 4-16
DISKETTE AND HARD DISK ERRORS 4-17
DUAL-MODE FUNCTION KEYS USED FOR COMMAND ENTRY 4-18
RULES FOR ENTERING COMMANDS 4-20
MS-DOS COMMAND DESCRIPTIONS 4-20
  BREAK Command 4-21
  CAT Command 4-22
  CHDIR Command 4-23
  CHKDSK Command 4-24
  CHR Command 4-27
  CIPHER Command 4-30
  CLS Command 4-31
  COPY Command 4-32
  CTTY Command 4-34
  DATE Command 4-35
  DEL Command 4-37
  DIR Command 4-38
  DISKCOPY Command 4-39
  DISP Command 4-41
  DISP1 Command 4-42
  ECHO Command 4-43
  EXIT Command 4-44
  FIND Command 4-45
  FOR Command 4-47
  FORMAT Command 4-48
  GOTO Command 4-49
  HDFORMAT Command 4-50
    B(ad Tracks 4-53
    V(olumes 4-54
    F(ormat 4-57
    E(xit 4-58
    U(pdate 4-58

Contents (cont'd)

Page

IF Command 4-59
KEY Command 4-60
LOCATE Command 4-63
MKDIR Command 4-65
MORE Command 4-66
PATH Command 4-67
PAUSE Command 4-68
POW Command 4-69
PRINT Command 4-70
PROMPT Command 4-73
RDCPM Command 4-75
RECOVER Command 4-76
REM Command 4-77
REN Command 4-78
RMDIR Command 4-79
SET Command 4-80
SETCOM Command 4-81
SHIFT Command 4-83
SORT Command 4-84
SYS Command 4-86
TIME Command 4-88
TIMEROFF Command 4-89
TIMERON Command 4-90
TYPE Command 4-91
VER Command 4-92
VERIFY Command 4-93
VOL Command 4-94

Chapter 5 The Line Editor (EDLIN)

HOW TO START EDLIN 5-1
EDITING KEYS USED BY EDLIN 5-3
  COPY1 5-4
  COPYUP 5-5
  COPYALL 5-6
  SKIP1 5-7
  SKIPUP 5-8
  VOID 5-9

Contents (cont'd)

Page

INSERT 5-10
REPLACE 5-11
NEWLINE 5-12
EDLIN COMMANDS 5-13
Information for Entering EDLIN Commands 5-13
Command Options 5-15
EDLIN COMMAND DESCRIPTIONS 5-16
Append 5-16
Copy 5-17
Delete 5-20
Edit 5-23
End 5-25
Insert 5-26
List 5-30
Move 5-33
Page 5-34
Quit 5-35
Replace 5-36
Search 5-39
Transfer 5-42
Write 5-43
ERROR MESSAGES 5-44

Appendix A Files Supplied on the MS-DOS Diskette

Illustrations

Figure	Title	Page
1-1	MS-DOS Loaded into Memory	1-3
2-1	The APC Keyboard	2-6
3-1	A Sample Hierarchical Directory Structure	3-10
4-1	How MS-DOS Uses the AUTOEXEC.BAT File	4-11

Tables

Table	Title	Page
2-1	Single Key Functions	2-7
2-2	Combined Key Functions	2-8
3-1	MS-DOS File Types	3-5
3-2	Device Names	3-5
4-1	Dual-Mode Function Keys Used in Command Line Editing	4-4
4-2	Control Character Functions	4-8
4-3	Dual-Mode Function Keys Used for Command Entry	4-19
4-4	CHKDSK Error Messages	4-25
4-5	CHR Sub-Commands	4-28
4-6	FIND Error Messages	4-46
4-7	KEY Sub-Commands	4-61
4-8	LOCATE Error Messages	4-64
4-9	PRINT Error Messages	4-71
4-10	User Prompt Characters	4-73
4-11	SYS Error Messages	4-87
5-1	EDLIN Commands	5-14
5-2	EDLIN Error Messages	5-44
A-1	Files Supplied on the MS-DOS Diskette	A-1

Preface

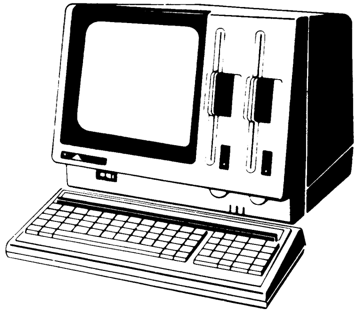
The *MS-DOS System User's Guide* is one of two documents published for users of Microsoft Corporation's MS™-DOS operating system as adapted for the Advanced Personal Computer (APC).

This guide serves as a tutorial and reference for the first time user. It contains a basic overview of MS-DOS. You are instructed on the initial installation of MS-DOS and introduced to the APC keyboard with its control key functions. You learn how program and data files are created and maintained. All the MS-DOS commands that execute system routines and utilities are described. Several of these commands provide access to functions that would be of more interest to programmers than business users. They are included in the guide because they contribute to a complete picture of how the MS-DOS commands work. Finally, the Line Editor utility (EDLIN) is presented. This utility is the one that will be used for recording and maintaining the information of a business.

The second document for the MS-DOS release is the *MS-DOS System Reference Guide*. This guide is divided into two parts. The first part discusses the characteristics of the MS-DOS operating system. The second part supplies information for program development under MS-DOS. For example, it describes the Microsoft Corporation's Macro Assembler™, the Linker, and other utilities for implementing special features, such as graphics displays. The additional information needed to use many of the commands presented in this guide can also be found in the *MS-DOS System Reference Guide*.

Chapter 1

Introduction to MS-DOS



MS™-DOS is an operating system designed by Microsoft Corporation that has been adapted for use on the Advanced Personal Computer (APC). This customized version of MS-DOS manages your computer system, while providing features that help you run your applications. Some of these features are

- Programmable keys on the APC keyboard to which you can assign special functions
- A method of organizing files by directories and subdirectories for easy location of information
- Programs to enhance your applications, such as the CHR utility that allows the display of specially created characters on the APC screen.

WHAT IS AN OPERATING SYSTEM?

If you are just beginning to use your APC, you may not know what an operating system is. An operating system can generally be described as a number of programs, or software, that control all the components of a computer system, both the equipment, or hardware, and other software running in it. Common tasks for an operating system to perform are the allocation of memory for executing programs and the placement and retrieval of data on storage devices (for the APC, these devices are diskettes and hard disks).

WHAT IS MS-DOS?

The MS-DOS operating system consists of three programs. Each time you use your APC, you must load, or copy, these programs into memory from the MS-DOS diskette provided to you. Once in memory, the MS-DOS programs begin their supervisory functions.

To run your applications or use any feature of your operating system, you enter an instruction to MS-DOS by typing a command at the APC keyboard. MS-DOS then performs the operations required to accomplish the task.

For the most part, MS-DOS operates automatically. After you give it the initial command, you do not need to issue any more instructions for completing a task. However, you will want to know how your operating system works in general to make the best use of its features. An overview of MS-DOS is presented in the following sections of this chapter.

The MS-DOS Programs

MS-DOS runs by the combined operations of its three programs. These programs can be briefly described as follows:

> COMMAND.COM — is the command processor that interprets the commands you enter to MS-DOS.
>
> IO.SYS — is the program that controls all the peripheral devices attached to your system. IO.SYS is active whenever you use such a device, be it the keyboard to type in commands, the screen to display information, or a diskette drive to store data.
>
> MS-DOS.SYS — is the heart of MS-DOS. It directs major system operations, like placing programs to be executed into memory. MS-DOS.SYS is constantly "interfacing" with COMMAND.COM and IO.SYS, giving these programs instructions to perform some function and regaining control after their operations have completed.

How MS-DOS Manages System Resources

When the MS-DOS programs are loaded, they are placed in a part of memory referred to as system memory. From here, these programs manage the resources of your computer system: memory, diskette and hard disk drives, the printer, the APC screen, keyboard, and any auxiliary devices like data communications lines.

Figure 1-1 represents memory after MS-DOS has been loaded.

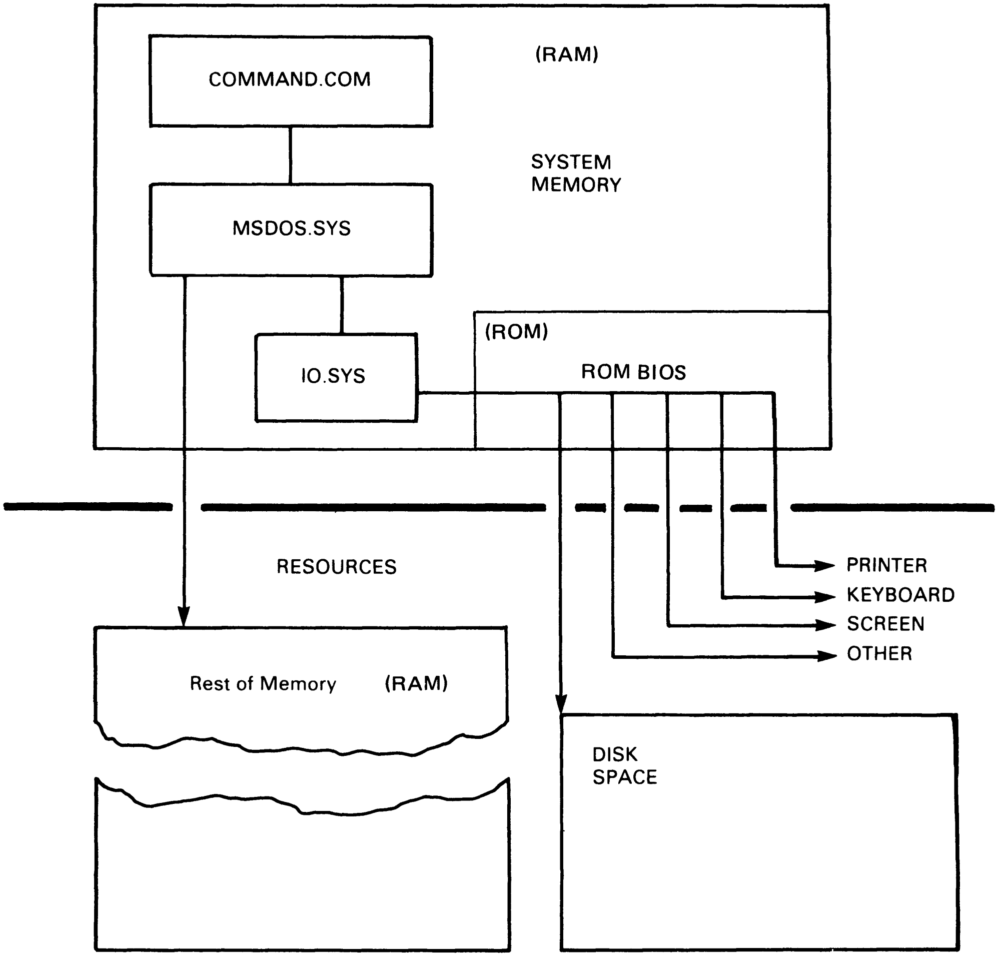


Figure 1-1 MS-DOS Loaded into Memory

In Figure 1-1, the memory area not occupied by the MS-DOS programs is used to hold the applications programs which run under MS-DOS while they execute. The working storage areas used by these programs are also reserved here.

After the MS-DOS programs are loaded into memory, they usually remain there until you shut off your APC or until a problem occurs that causes you to reload your operating system. Sometimes, however, the memory requirements for executing a large program cause MS-DOS to overlay the area where the command processor resides. If this happens, MS-DOS automatically reloads the part of the command processor overlaid, as long as the diskette on which this program is stored is still in a diskette drive. If you have removed the diskette, you will get the following message on your APC screen:

```
Insert MS-DOS diskette in default drive
and strike any key when ready
```

Insert the MS-DOS or other diskette with copies of the MS-DOS programs on it in the default drive (the diskette drive that MS-DOS accesses automatically) for reloading the command processor.

If MS-DOS finds an incorrect version of the command processor on the diskette, the following message is displayed:

```
Invalid COMMAND.COM
Insert COMMAND.COM diskette in default drive
and strike any key when ready
```

As instructed, replace the diskette in the default drive with one containing a correct version of the command processor and press any key to continue.

MEMORY

MS-DOS's main function in controlling memory is the allocation of space for programs to execute. MS-DOS may reserve one area of memory for a program, or increase and decrease the amount of memory used by a program as it runs. It can also swap parts of a large program that is executing in and out of memory to manage this resource most efficiently. Programs that require a lot of memory and, thus, are swapable are distinguished by their names. You will learn about program, or executable, file names in Chapter 3 of this guide.

DISKETTE DRIVES

Depending on the configuration of your APC system, you have one to four diskette drives. The APC diskette drives use 8-inch diskettes (in other systems, these storage media are called floppy disks or just floppies). These diskettes can be single-sided, single-density or double-sided, double-density, the latter providing up to 1,250,304 bytes (one megabyte) of storage space. You can store programs and data to be processed on these diskettes in files that you create.

When you create files on a diskette, you do not have to set up individual files. As part of a feature called its "hierarchical directory structure," MS-DOS assigns the file to a directory, or a group of files. The first directory on a disk is called the root directory and it is through this file that you locate all the others on the diskette. In addition to the root directory, you can have any number of subdirectories. All the directories form a hierarchical structure, any level of which can be accessed with commands either directly or indirectly by tracing through the upper levels. Details on file creation and the hierarchical directory structure are given in Chapter 3 of this guide.

HARD DISK DRIVES

MS-DOS can also manage up to two hard disk drives. Using the HDFORMAT utility, you can divide the storage space on these drives into "volumes," or units of space, of various sizes. These "logical," as distinguished from physical, volumes can be "mounted" onto "logical drives" in much the same manner as diskettes are inserted into diskette drives. MS-DOS provides four logical hard disk drives designated by the letters E, F, G, and H. Hard disk volumes can be any size from very small (smaller than the size of a diskette) to very large (the entire hard disk can be designated as a single logical volume).

Files are created on hard disk in the same manner that they are created on diskette, except that root directories are assigned to each logical volume on the device. For details on file creation for hard disk, see Chapter 3.

THE PRINTER

One of two NEC printers can be a part of your computer system. The 8023 Dot Matrix Printer prints 100 characters per second with up to 136 characters on a line of paper. The Spinwriter Model 3530 produces text in fully-formed type face at 35 characters per second.

When you use your printer for output from a program, MS-DOS directs the operation as though you were calling to use a file. That is, when you want to use the printer you direct MS-DOS to do so by entering a command containing the printer's "filename."

This method of accessing, or addressing, the printer is called "device independent I/O." It gives you greater flexibility in using devices for input and output. For example, programs designed originally only for diskette file I/O can have their input come from the keyboard or their output sent to the printer by entering the appropriate command.

How you enter commands to use the printer is discussed in Chapter 3 in the section on filenames.

THE APC SCREEN

Your APC console (the processor unit and attached keyboard) has a screen that allows for the display of text or graphics in either monochrome or color. The entire screen holds up to 25 lines of 80 characters each (or 2000 characters). With commands, you can turn various MS-DOS functions on and off, including the displaying of the date and time and special characters. In the same way as for the printer, you can send output to the screen by entering the screen's "filename" in a command. See Chapter 3 for an explanation of how to do this.

AUXILIARY DEVICES

MS-DOS controls the operations of the auxiliary devices used in your computer system. An example of such a device is a serial communication line for data communications. This guide does not provide information on the installation of auxiliary devices or the software required to use them. For this, you should consult your APC representative. However, because MS-DOS does support data communications through an RS-232-C interface, the command to define the working parameters for this facility is included in the MS-DOS commands in Chapter 4.

SOFTWARE PROVIDED ON YOUR MS-DOS DISKETTE

Your MS-DOS diskette contains the following files:

the MS-DOS system programs and required system information files,

the programs executed by the MS-DOS external commands (those that must be loaded into memory before executing),

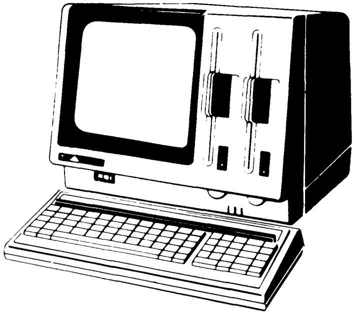
several programs to use special features of the APC, such as programmable function keys and alternate character sets, and

the Macro Assembler™, Microsoft's assembler.

For a complete listing of the files on the MS-DOS diskette, refer to Table A-1 in Appendix A.

Chapter 2

Getting MS-DOS Started



Whether you are starting up your MS-DOS operating system for the first time, or preparing to run your applications for the day, you begin the same way.

To use your MS-DOS operating system, you must load, or read a copy of it into the memory of your APC.

First, turn on the power at the console and on the disk drives, if you have any. Insert your MS-DOS diskette into drive A and close the drive door. This automatically loads MS-DOS into memory (the symbols — [LOD] and [LOD M] — appear on the screen in acknowledgement).

THE APC STATUS LINE

When MS-DOS is loaded, the APC status line appears on the first line of your screen. It is 80 characters long and looks as follows:

```
MS-DOS version v2.0(000) CAP ALT GR1 GR2 sp:9 Thu 06/30/83 22:10
```

At the left of the line are the version and revision numbers of the MS-DOS software.

The names of the latching keys on the APC keyboard — CAPS LOCK, ALT, GRPH 1, and GRPH 2 — are represented to show which ones are active. When any of these keys are being used, their names are displayed in reverse video (in black on green).

The current screen display speed, set using the CTRL key and the numbers on the typewriter keyboard, is also given. In the example above, the display speed is represented by sp:9.

The day of the week, system time, and system date end the APC status line information. These display items can be set using the TIME and DATE commands (see Chapter 4).

MS-DOS VERSION NUMBER AND TIME AND DATE PROMPTS

Once the command processor is loaded, you will see the following display on your screen. The underscore represents the cursor. The cursor is a blinking block that shows the first character position on the screen where you can type a letter or other character.

```
MS-DOS Version 2.00
Copyright 1981, 82, 83 Microsoft Corp.

Command V. 2.02
Current date is Thu 6-30-1983
Enter new date: _
```

You must now enter today's date and time. In this guide, the term "enter" designates typing information then pressing the RETURN key to register your response in the computer.

If the displayed date is correct, press RETURN. From here on, when you are instructed to use any key or key combination, their names will appear in bold type face. Otherwise type the date in an mm-dd-yy format, where:

mm is a one- or two-digit number from 1-12
(representing month)

dd is a one- or two-digit number from 1-31
(representing day of month)

yy is a two-digit number from 80-99 (the 19 is assumed), or a four-digit number from 1980-2099
(representing year)

Separators between the numbers in the date can be hyphens (-) or slashes (/). For example,

8-1-83 or 08/01/83

Both are acceptable answers to the Enter new date: prompt.

If you enter an invalid date or form of date, the system will prompt you again with "Enter new date:".

> NOTE
>
> If you make a mistake while typing, press DEL. You can then re-answer the prompt.
>
> Another way to correct an entry is to use the BACK SPACE key. BACK SPACE moves the cursor backwards, erasing one letter at a time.

After you enter the date you will see a prompt similar to this:

```
Current time is 8:30:14.32
Enter new time: _
```

If the displayed time is correct, press RETURN. If not, enter the time in the hh:mm format, where:

hh	is a one- or two-digit number from 0-23 (representing hours)
mm	is a one- or two-digit number from 0-59 (representing minutes)

MS-DOS uses this time value to keep track of when you last updated and/or created files on the system. Notice that MS-DOS uses military time; for instance, 1:30 p.m. is written 13:30. For example,

```
Current time is 8:22:14.32
Enter new time: 9:05
```

You should use the colon (:) only to separate hours and minutes. If you enter an invalid number separator, MS-DOS will repeat the prompt.

THE MS-DOS SYSTEM PROMPT

After you have answered the new time prompt, a message is displayed that looks like this:

```
A> _
```

The A> is the system prompt from the command processor. It tells you that MS-DOS is ready to accept commands. The A in the prompt represents the default drive. This means that MS-DOS will search only the diskette in drive A for any filenames you may enter and will write files to that diskette unless you specify a different drive in your command. (More information on the default drive is given in the section on drive designators and the default drive later in this chapter.)

How to specify a different default drive is explained later in the section on drive designators and the default drive designation.

ENTERING COMMANDS TO MS-DOS

When the MS-DOS system prompt is displayed, you can enter a command to perform some task.

Command Types

There are two types of MS-DOS commands: internal and external.

Internal commands execute programs that have been included in the command processor. Because these programs are loaded into memory at system startup, they do not have to be read into memory from the MS-DOS diskette (or other diskette or hard disk on which they are stored) before they are executed.

External commands, on the other hand, are commands that execute programs not automatically loaded at system startup. Each time that you run one of the programs executed by an external command, it must be read into memory from the diskette or hard disk where it is stored.

Ordinarily, you will not be concerned about the type of command you are entering. However, in the case of external commands, you should make sure that the program file to be executed is on the specified hard disk or diskette. If the program is stored on a particular diskette, for example, that diskette should be inserted in a diskette drive.

For detailed descriptions of the internal and external commands, see Chapter 4.

MS-DOS Command Format

Commands that you enter to MS-DOS must be typed in a specific format. This format can be described as follows:

```
Command [options...]
```

where "options" represents a drive letter, the name of the file operated on, and other information required to perform a task. Chapter 4 describes the specific command options in detail.

The Command Line

When you type a command at the system prompt, MS-DOS reserves 127 character positions on the screen for your entry. These character positions constitute the command line. The command line begins right after the > sin the system prompt, goes to the end of the line on the screen, and then continues over onto the next line.

When you enter characters on the command line, MS-DOS stores them in a storage area in memory called the template. The template feature allows you to retrieve your last-entered command line, edit it, and use it again. For a full description of the template, refer to Chapter 4.

To enter a command on the command line, you begin typing right after the > in the system prompt (MS-DOS also allows spaces between the prompt and your command).

You can type the command in any combination of uppercase and lowercase letters. MS-DOS treats all letters in the command line as uppercase.

For a demonstration, enter the internal command, DIR, at the system prompt.

```
A> DIR
```

This entry tells MS-DOS to display the names of the files on the diskette in drive A on the APC screen. Since you have your MS-DOS diskette in this drive, you will get a listing of the files on that diskette. The display will be similar to the following:

```
Volume in drive A has no label
Directory of A:

COMMAND   COM   15480   2-08-83    7:50p
DEBUG     COM   11764   2-01-83   10:13a
CHKDSK    COM    6330   2-01-83    9:16a
EDLIN     COM    4389   2-01-83    9:31a
PRINT     COM    3808   2-01-83   12:39p
RECOVER   COM    2277   2-01-83    2:22p
                    .
                    .
                    .
```

The information given for a file is the filename (a combination of a unique first part and a filename extension — for example, COM — identifying the file type), the size of the file in bytes, and the date and time it was last used.

MS-DOS recognizes only correctly spelled commands. If you enter one incorrectly, MS-DOS displays an error message for you to make another entry.

```
A> DJR
Bad command or file name
```

THE APC KEYBOARD

Before you attempt to load MS-DOS, you should familiarize yourself with the APC keyboard. Figure 2-1 is an illustration of the keyboard.

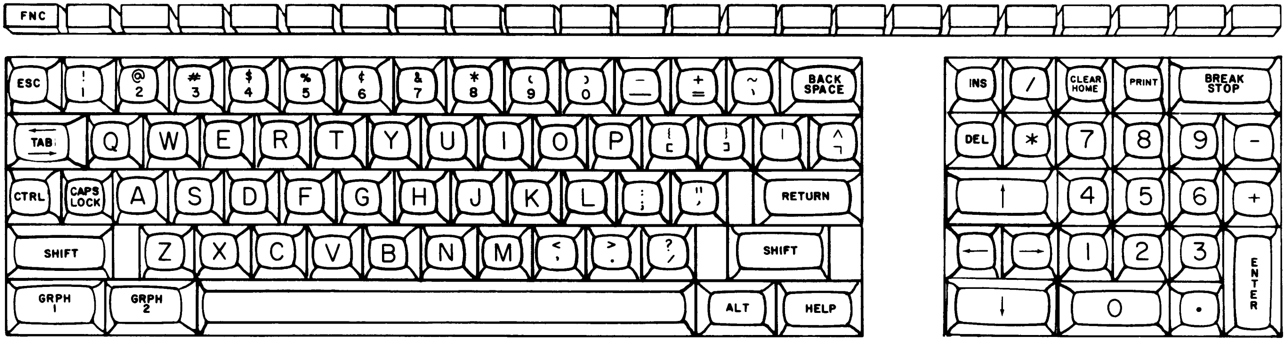


Figure 2-1 The APC Keyboard

The APC keyboard consists of three areas of keys:

A typewriter-like keyboard

A numeric keypad with some standard function keys

Twenty-two dual-mode function keys

A few features enhance command entry and program execution:

- Programmable function keys — Using the KEY utility, you can define up to 32 different operations to be performed by the dual-mode function keys at the top of the keyboard. Any number of key definition sets can be stored in files and used where applicable.

- Repeat function — Instead of repeatedly pressing a particular key, you can enter a string of characters by holding down the key. The character is repeated until you release the key.
- Key clicker — There is a "key clicker" on the APC that can be enabled to produce a brief tone when keys are pressed. This feature is called the SOUND BEEP function. Refer to the *MS-DOS System Reference Guide* for more information.
- FIFO buffer — MS-DOS also provides a FIFO buffer (standard storage area) for the APC keyboard. This buffer recognizes and remembers the keys pressed in first-in, first-out sequence. This allows you to type up to 64 characters ahead while other processing occurs. (The FIFO buffer should not be confused with the command line buffer, or template. The FIFO buffer facilitates the system's handling of key commands.)

In most cases, the keys on the APC keyboard perform as they are labeled. Some keys, such as HELP and INS, generate special codes that can be "trapped" (sensed) and used in application programs to perform special routines. The dual-mode function keys, PF1-PF22, can be assigned values or functions in individual applications. The ALT key is a latching key (it remains pressed down until released) that allows application programs to trap additional character codes from the keyboard. Still other keys, such as those listed in Table 2-1, perform the functions described.

Table 2-1 Single Key Functions

KEY	FUNCTION
DEL	Effectively deletes all the characters in a command line by erasing them from the template. On the screen, a slash is displayed after the command entry and the cursor moves to the beginning of the command line.
PRINT	Puts the printer online to echo all entries made from the keyboard by printing them out; pressing this key again takes the printer offline.
BREAK STOP	Temporarily stops console listing; pressing again resumes the listing.
BACK SPACE	Moves the cursor back one space, erasing the character positioned there.
TAB	Moves the cursor to the next tab stop.

At other times the keys on the typewriter keyboard and numeric keypad can be used in combinations that perform unique functions. Table 2-2 describes some of these combined key functions. The functions are performed by pressing the mode key and the effective key or keys simultaneously.

Table 2-2 Combined Key Functions

MODE KEY	EFFECTIVE KEYS	FUNCTION OR ACTION
CTRL	0-9 on the typewriter keyboard	Controls the speed at which information is displayed on the screen. CTRL-0 is the slowest speed; CTRL-9 is the fastest.
		Scrolls the display up and down.
	A-Z, @, [,\,], ^ , ¯	Generates ASCII control codes.
FNC	CTRL + BREAK STOP	Pressed simultaneously, these three keys reload MS-DOS.
	PF1-PF16	Provides additional functions through key use.
CAPS LOCK	A-Z on the typewriter keyboard	Allows typing of uppercase letters only. Note that neither the numbers on the typewriter keyboard nor those on the numeric keypad are affected.
SHIFT	All keys on the typewriter keyboard	Types the top symbol on each key pressed.
	CLEAR HOME	Clears the screen.
	BREAK STOP	Generally, aborts the program that is currently running.
GRPH 1	All typewriter keys	Used with and without the SHIFT key to allow access to graphic symbols and characters.
GRPH 2		Same function as for GRPH 1.

DRIVE DESIGNATORS AND THE DEFAULT DRIVE

As you learned in Chapter 1, the APC can have up to four diskette drives and four "logical" hard disk drives. Each of these devices is identified to MS-DOS by a letter called a drive designator. For example, the drive designator for the first diskette drive in a system is A. If there are more diskette drives, the drive designators are B, C, and D for the second, third, and fourth devices. If your APC is equipped with a hard disk drive, then the "logical" drive designators are E, F, G, and H.

MS-DOS treats one of the drives, called the default drive, in a special manner. The default drive is indicated by the letter appearing in the system prompt.

After you load MS-DOS, for instance, the default drive is A.

```
A>_
```

The A in the system prompt means that the first diskette drive is the device where MS-DOS is to look for any programs or files required to execute a command. MS-DOS performs this automatic search operation, unless you specify a different device to search in your command.

For example, you were asked to enter DIR at the system prompt the first time you saw it on the screen.

```
A>DIR
```

In response, MS-DOS displayed the names of the files in the root directory of the diskette in drive A.

In order to display the directory information on a device other than the default device, you must direct MS-DOS specifically to search that device. For example, to get a display of the root directory on the diskette in diskette drive B, enter

```
A>DIR B:
```

By specifying B in the command above, you instructed MS-DOS to search drive B for the root directory and to display the information.

CHANGING THE DEFAULT DRIVE DESIGNATION

Occasionally, you will want to change the default drive designation to another drive in your system. Perhaps you have a series of programs to run that are all stored on the same device and this device is other than the current default device. To execute the programs one after the other without having to enter the device designator in the command for each program, you can change the default drive designation.

For example, to change the default drive from A to B, you enter the drive designator B followed by a colon (:).

```
A> B:
```

MS-DOS returns with

```
B>_
```

The system prompt has been changed to indicate drive B as the default drive. Now, if you enter DIR alone in response to the system prompt, you will get a directory display for the diskette in drive B.

Note that MS-DOS now assumes that B is the default drive and will perform its automatic search operation only on that device until you specify a different one.

You can change the default drive designation to any of the diskette or hard disk drives in your system.

Example:

```
B> E:
```

This entry changes the default drive designation from the diskette drive B to logical hard disk drive E.

SINGLE-DRIVE SYSTEMS

Some APC systems only have a single diskette drive. If your system is of this type, you can still access a second "logical" diskette drive (B). This capability is important. It permits you to do operations that require two diskette drives, such as copying files from one diskette to another.

Think of your system as having two diskette drives (drive A and drive B). But instead of A and B representing two physical drives as in a multi-drive system (a system with two or more diskette drives), the drive designators represent diskettes.

When MS-DOS performs operations that require two diskette drives in a single-drive system, it prompts you to swap the "A" and "B" diskettes in the single drive whenever it needs to access a file on a different diskette.

For example, to copy the COMMAND.COM file from the MS-DOS diskette to another, you enter

```
A>COMMAND.COM B:
```

MS-DOS responds with

```
Insert disk for drive B:
and strike any key to continue
```

At this point, you remove the MS-DOS diskette from drive A and replace it with the diskette to which COMMAND.COM is to be copied. Next, you press any key for the COPY operation to continue.

What actually happened was that MS-DOS moved the COMMAND.COM file into memory from the "A" diskette, then wrote this file onto the "B" diskette after you changed diskettes and pressed a key.

The COPY operation ends with the following message being displayed:

```
1 File(s) copied
```

Note that for large files that cannot fit into memory all at once, you will have to repeat the "swap" procedure a number of times until the entire file has been copied.

MAKING A WORKING COPY OF THE MS-DOS DISKETTE

One of the first tasks you should perform after receiving your MS-DOS diskette is to make a working copy. This procedure ensures that you will always have a good copy of the MS-DOS diskette in the case that the working copy gets damaged.

Making a copy of a diskette involves two steps: formatting the diskette to receive the copied information and then copying the information.

Formatting a Diskette

You must format, or prepare, all new diskettes before they can be used by MS-DOS.

A blank diskette is formatted using the FORMAT command. This command changes the diskette to a format that MS-DOS can use; it also analyzes the diskette for defective tracks (data storage areas). If the diskette is not already blank, formatting it will destroy any data that exists on the diskette. Although this can be a convenient way to make a blank diskette, it is recommended that the DELETE command be used for removing unwanted files. Refer to Chapter 4 for more information on the DELETE command.

The syntax of, or entries in proper order for, the FORMAT command are

```
FORMAT [d:]
```

where:

d: is the drive designation (for example, A or B in a two-drive system), or the drive that contains the diskette to be formatted.

Note that the brackets ([]) identify optional information. However, if you do not specify a diskette drive, MS-DOS will format the diskette in the default drive.

WARNING

> You are about to format a diskette which you will use to make a copy of your MS-DOS diskette.
>
> When you enter the FORMAT command to format this diskette, be sure to specify B as the drive designation in the command. If you omit the drive designation, you will accidentally format your MS-DOS diskette in drive A. If this happens, you will no longer be able to use your MS-DOS diskette.

If the FORMAT utility is on your MS-DOS diskette and that diskette is already in drive A, you can proceed to format your new blank diskette. The following command will format the diskette in drive B.

```
FORMAT B:
```

MS-DOS issues the following message:

```
Insert new diskette for drive B:
and strike any key when ready
```

Before inserting the "drive B diskette," look at it to see if it has a notch cut out of the lower right hand edge. If there is a notch, the disk is "write protected," that is, the notch will prevent data from being written to the diskette. Since formatting involves writting on the diskette, you must "write enable" it. To do this, place a small foil tab over the write-protect notch. These tabs are supplied with the diskettes.

After you insert the diskette to be formatted in drive B, press any key on the keyboard. You can now use the DISKCOPY command to transfer the information on your MS-DOS diskette to this diskette. Before you do this though, you should read a little more about the FORMAT command.

FORMAT prepares a diskette for the storage of files by placing on it certain system information required to access the files. This system information consists of an empty root directory and File Allocation Table (FAT) that will contain the file locations when files are actually stored on the diskette. For more information on root directories and FATs, see Chapter 3.

From time to time, you will want to format a diskette that is "bootable," that is, a diskette that can load MS-DOS as well as contain program and data files. To make a diskette bootable, you must make a copy of the MS-DOS programs on it. To copy these programs from your MS-DOS diskette to another diskette, you specify an option, /S, in the FORMAT command.

```
FORMAT B: /S /V
```

Note that the command contains another entry -- /V. The /V option specifies a volume label for a diskette. Volume labels serve as names by which you can identify diskettes. To display the volume label for a diskette, you use the VOL command (see Chapter 4 for a description of this command).

When the formatting is finished, MS-DOS will issue a message similar to this:

```
System transferred
Volume label (11 characters, ENTER for none)?
```

Type a volume label in response to the above prompt if you want to name the diskette, and press RETURN. An example of a volume label is PROGRAMS. If you do not want to attach a label to a diskette, simply press RETURN. You will see on your screen a message similar to this:

```
    1250304 bytes total diskette space
    54272 bytes used by system
    1196032 bytes available on diskette
Format another (Y/N)?_
```

Enter Y to format another diskette. Enter N to end the FORMAT program.

Copying from Diskette to Diskette

The DISKCOPY command copies the contents of one diskette onto another. DISKCOPY is the fastest way of copying a diskette because it copies the entire diskette in one operation, including MS-DOS programs if they exist.

The format of the DISKCOPY command is

```
DISKCOPY [drive1:] [drive2:]
```

Drivel is the diskette drive that contains the diskette that you want to copy; drive2 is the diskette drive that contains the blank or "destination" diskette. The blank diskette must be formatted prior to running DISKCOPY.

To make a copy of your MS-DOS diskette which is in drive A, enter

```
DISKCOPY A: B:
```

MS-DOS responds

```
Insert SOURCE diskette into drive A
Insert for malted target diskette into drive B
Press any key when ready
```

Make sure the MS-DOS diskette is in drive A and insert a blank, formatted diskette in drive B. Press any character key after you have done this and MS-DOS will begin copying the MS-DOS diskette. After MS-DOS has copied the diskette, it prompts you

```
Copy another (Y/N)?
```

Enter N if you do not want to make another copy of your MS-DOS diskette. The system prompt is redisplayed.

You now have a duplicate copy of your MS-DOS diskette in drive B. This duplicate copy can be used as a working copy and your MS-DOS diskette saved as a backup copy.

Diskettes must be the same size and density to be copied with the DISKCOPY command. If either of the diskettes that you are using has defective tracks, DISKCOPY will not work. Use the COPY command to back up your diskettes in these cases. (COPY will skip over defective tracks.) Refer to Chapter 4 for information on how to use COPY to back up your diskettes. Chapter 4 also gives more information on the DISKCOPY command.

Note that DISKCOPY cannot be used for copying hard disk volumes. If you want to copy individual files or directories to or from hard disk, use the COPY command.

AUTOMATIC PROGRAM EXECUTION

If you want to run a specific program automatically each time you start MS-DOS, you can do so with the automatic program execution feature. For example, you may want to have MS-DOS display the names of your files each time you load MS-DOS.

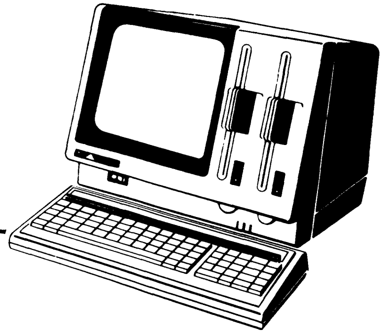
When you start MS-DOS, the command processor searches for a file named AUTOEXEC.BAT on the MS-DOS diskette. This file contains commands that MS-DOS will execute each time MS-DOS is started. Chapter 4 tells you how to create an AUTOEXEC.BAT file.

HOW TO TURN THE SYSTEM OFF

There is no "logoff" command in MS-DOS. To end your computer session, open the diskette drive doors and remove the diskettes. Then, simply turn your APC off in response to the system prompt.

NOTE

> Always remove the diskettes from the diskette drives before you turn off the APC.



Chapter 3

MS-DOS Files

In Chapter 1, you learned that directories contain the names of your files. In this chapter, you will learn how to name and copy your files. You will also learn more about the MS-DOS hierarchical directory structure that makes it easy for you to organize and locate your files.

WHAT IS A FILE?

An MS-DOS file is a collection of information stored on a diskette or hard disk.

There are two kinds of files: program files and data files. A program file is an executable file, that is, a series of instructions that MS-DOS can follow step by step. Examples of executable files are DISKCOPY, the diskette copy utility, and AUTOEXEC.BAT that contains MS-DOS commands for performing a task automatically at system startup. A data file contains information, such as a list of customer names and addresses or the text of a document. This information is not "executable," but can be processed by programs as they execute. Data files can also store the source code for a program. A program source file must be processed by a language compiler or assembler to convert it into the object code, or instructions, which will be executed. Usually, an executing program processes a data file. However, some times a program will execute other executable program files. For example, the internal command EXIT executes a program that stops the running of an application program by calling in the command processor for another task.

HOW ARE FILES CREATED?

Data files can be created using the Line Editor utility (EDLIN) described in Chapter 5. This utility allows you to enter text line by line, then to modify it using several single and combined keystrokes. Also, application programs will often create files.

The Macro Assembler produces output files of executable code from input files of source code.

Finally, you can create a file by using the COPY command to make a copy of it on the same or a different storage medium.

FORMATTING DISKETTES AND HARD DISKS BEFORE CREATING FILES

Before you create files on either diskette or hard disk, you must format, or prepare, the storage medium. Formatting for either diskette or hard disk involves running a utility that places the system information required to access files on it and checks for defective tracks.

You have already used the diskette formatting utility, FORMAT, to prepare a diskette for making a copy of your MS-DOS diskette (see Chapter 2). Complete instructions for using this utility are found in Chapter 4 under the FORMAT command.

To format a hard disk, you use the hard disk formatting utility, HDFORMAT.EXE. In addition to placing required system information on the hard disk and checking for defective tracks, HDFORMAT.EXE partitions, or divides, the storage space into areas called volumes. Volumes are approximately the same size in storage capacity as a double-sided, double-density diskette. When you read input from or send output to a file on hard disk, MS-DOS searches for the volume identifier as well as the name of the file being accessed. For details on how to use HDFORMAT.EXE, refer to the description of the HDFORMAT command in Chapter 4.

HOW MS-DOS KEEPS TRACK OF YOUR FILES

The names of files are kept in directories on a diskette or hard disk. These directories also contain information on the size of the files, their location on the storage medium, and the dates that they were created and updated. The directory you are working in is called your current or working directory.

An additional system area is called the File Allocation Table. It keeps track of the location of your files on the diskette or hard disk. It also records the location of free, or available, space which can be added to existing files or allocated to new files.

These two system areas, the directories and the File Allocation Table, enable MS-DOS to recognize and organize the files on your diskettes and hard disks. The File Allocation Table is copied onto a new diskette or hard disk volume when you format it with the FORMAT command and one empty directory is created, called the root directory.

The DIR (Show Directory) Command

If you want to know what files are on your diskette or hard disk, you can use the DIR command. This command tells MS-DOS to display all the files in the current directory on the storage device that is named. For example, if your MS-DOS diskette is in drive A and you want to see the listing for the current directory on that diskette, enter

```
DIR A:
```

MS-DOS will respond with a directory listing of all the files in the current directory on your MS-DOS diskette. The display should look similar to this:

```
Volume in drive A has no label
Directory of A:

COMMAND    COM   15480   2-08-83    7:50p
DEBUG      COM   11764   2-01-83   10:13a
CHKDSK     COM    6330   2-01-83    9:16a
EDLIN      COM    4389   2-01-83    9:31a
PRINT      COM    3808   2-01-83   12:39p
                   .
                   .
                   .
RECOVER    COM    2277   2-01-83    2:22p
          41 File(s)    513024 bytes free
```

Note that DIR also indicates the number of files in the directory as well as the amount of free space left on the diskette (in bytes).

You can also get information about a file on your diskette by typing DIR and its filename. For example, if you have created a file named MYFILE.TXT, the command

```
DIR MYFILE.TXT
```

will give you a display of all the directory information (name of file, size of file, date last edited) for the file MYFILE.TXT.

What you do not see in the above directory listing for your MS-DOS diskette are the names of the MS-DOS programs MSDOS.SYS and IO.SYS. These programs are present on the diskette; however, they are "hidden" for protection and not viewable using DIR. For more information on the DIR command, refer to Chapter 4.

The CHKDSK (Check Disk) Command

The CHKDSK command is used to check your diskettes for consistency. CHKDSK analyzes the directories and the File Allocation Table on the diskette that you specify. It then produces a status report of any inconsistencies, such as files which have a non-zero size in their directory but really have no data in them.

To check the diskette in drive A, enter

```
CHKDSK A:
```

MS-DOS will display a status report and any errors that it has found. An example of this display and more information on CHKDSK can be found in the description of the CHRDSK command in Chapter 4. You should run CHKDSK occasionally for each diskette to ensure the integrity of your files.

HOW TO NAME YOUR FILES

A typical MS-DOS file name will look like this one:

```
NEWFILE.EXE
```

The name of a file consists of two parts: the filename and the filename extension. In the example above, the filename is NEWFILE and the filename extension is EXE.

A filename can be from 1 to 8 characters long. The filename extension can be three or fewer characters. You can type any filename in small or capital letters and MS-DOS will translate these letters into uppercase characters.

Your filenames will probably be made up of letters and numbers, but other characters are allowed, too. Legal characters for filename extensions are the same as those for filenames. Here is a complete list of the characters you can use in filenames and extensions:

```
A-Z 0-9 $ & # % ' () - @ \ ^ { }  ~ ` !
```

Although you can designate any three legal characters as a file extension, you should know that MS-DOS reserves certain ones to represent specific file types. Table 3-1 describes these file types.

Table 3-1 MS-DOS File Types

FILE TYPE	DESCRIPTION
EXE	A loadable, executable program that is relocatable (if larger than 64K bytes). A loadable, executable program that can execute in 64K bytes of memory or less.
ASM	MACRO-86 source file
SYS	MS-DOS system program file
$$$	Temporary file created by language compilers and certain utilities for work space.
BAK	A file used by EDLIN to save a copy of a file to be edited before any changes are made to it.
KEY	Programmable function key definition file
CHR	Auxiliary APC character set file

MS-DOS treats some device names specially, and certain three-letter names are reserved for the names of these devices. These three-letter names cannot be used as filenames or extensions. You must not give your files any of the names shown in Table 3-2.

Table 3-2 Device Names

NAME	DESCRIPTION
AUX	Used when referring to data input from or output to an auxiliary device, such as a printer or disk drive.
CON	Used when referring to keyboard input or to output to the APC screen (console).
PRN	Used when referring to the printer device.
NUL	Used when you do not want to create a particular file, but the command requires an input or output filename.

Even if you add device designations or filename extensions to these filenames, they remain associated with the devices listed above. For example, A:CON.XXX still refers to the screen and is not the name of a file.

In addition to the filename and the filename extension, the name of your file may include a drive designation. A drive designation tells MS-DOS to look on the diskette or hard disk to find the filename typed. For example, to find directory information about the file NEWFILE.EXE which is located on the diskette in drive A (and drive A is NOT the default drive), enter the following command:

```
DIR A:NEWFILE.EXE
```

Directory information about the file NEWFILE.EXE is now displayed on your screen.

If drive A is the default drive, MS-DOS will search only the diskette in drive A for the filename NEWFILE, so the drive designation is not necessary. A drive designation is needed only if you want to tell MS-DOS to look on the other drive to find a file.

All of the parts of a filename make up a file specification. The term file specification (or filespec) will be used in this manual to indicate the following filename format:

```
[<drive designation:>] filename [<.filename extension> ]
```

Remember that brackets indicate optional items. Angle brackets mean that you supply the text for the item. Note that the drive designation is not required unless you need to indicate to MS-DOS on which storage medium to search for a specific file. You do not always have to give your filename a filename extension. The MS-DOS commands and utilities supplied with MS-DOS that require entering a file extension, direct you to do so.

Examples of file specifications are

```
B:MYPROG.COB
A:YOURPROG.EXT
A:NEWFILE.
TEXT
```

WILD CARDS

Two special characters called wild cards, can be used in filenames and extensions: the asterisk (*) and the question mark (?). These characters give you greater flexibility when using filenames in MS-DOS commands.

The ? Wild Card

A question mark (?) in a filename or filename extension indicates that any character can occupy that position. For example, the command

```
DIR TEST?RUN.EXE
```

will list all directory entries on the default drive that have eight characters, begin with TEST, have any next character, end with the letters RUN, and have a filename extension of .EXE. Here are some examples of files that might be listed by the above DIR command:

```
TEST1RUN.EXE
TEST2RUN.EXE
TEST6RUN.EXE
```

The * Wild Card

An asterisk (*) in a filename or filename extension indicates that any character can occupy that position or any of the remaining positions in the filename or extension. For example,

```
DIR TEST*.EXE
```

will list all directory entries on the default drive with filenames that begin with the characters TEST and have an extension of .EXE. Here are some examples of files that might be listed by the above DIR command:

```
TEST1RUN.EXE
TEST2RUN.EXE
TEST6RUN.EXE
TESTALL.EXE
```

The wild card designation *.* refers to all files on the storage medium. Note that this can be very powerful and destructive when used in MS-DOS commands. For example, the command DEL *.* deletes all files on the default drive, regardless of filename or extension.

For example, to list the directory entries for all files named NEWFILE on drive A (regardless of their filename extensions), simply enter

```
DIR A:NEWFILE.*
```

Or, to list the directory entries for all files with filename extensions of .TXT (regardless of their filenames) on the diskette in drive B, enter

```
DIR B:????????.TXT
```

This command is useful if you have given all your text programs a filename extension of .TXT. By using the DIR command with the wild card characters, you can obtain a listing of all your text files even if you do not remember all of their filenames.

HOW TO COPY YOUR FILES

Just as with paper files, you often need more than one copy of a diskette file. The COPY command allows you to copy files from one diskette to another. You can also give the copy a different name if you specify the new name in the COPY command.

The COPY command can also make copies of files on the same diskette. In this case, you must supply MS-DOS with a different filename or you will overwrite the file. You cannot make a copy of a file on the same diskette unless you specify a different filename for the new copy.

The format of the COPY command is

```
COPY filespec [filespec]
```

For example,

```
COPY A:MYFILE.TXT B:MYFILE.TXT
```

will copy the file MYFILE.TXT on diskette A to a file named MYFILE.TXT on diskette B. A duplicate copy of MYFILE.TXT now exists.

If you want to duplicate the file named MYFILE.TXT on the same diskette, enter

```
COPY A:MYFILE.TXT A:NEWNAME.TXT
```

You now have two copies of your file on diskette A -- one named MYFILE.TXT and the other named NEWNAME.TXT.

You can also copy all files on a diskette to another diskette or hard disk volume (that is, make a backup copy) using COPY. Again, the same command can be used to copy hard disk files to diskette.

For a complete description of the COPY command, see Chapter 4.

HOW TO PROTECT YOUR FILES

As with any information system, errors may occur with MS-DOS and information may be misused. If you are processing information that cannot be replaced or requires a high level of security, you should take steps to insure that your data and programs are protected from accidental or unauthorized use, modification, or destruction. Simple measures you can take — such as removing your diskettes from drives when they are not in use, keeping backup copies of valuable information, and installing your equipment in a secure facility — can help you maintain the integrity of the information in your files. An MS-DOS command, CIPHER, can also be used to encrypt your files for total privacy. For more information on CIPHER, refer to Chapter 4.

DIRECTORIES

As you learned in Chapter 1, the names of your files are kept in a directory on each diskette and hard disk. The directory also contains information on the size of the files, their locations on the storage medium, and the dates that they were created and updated.

When there are multiple users on your computer, or when you are working on several different projects, the number of files in the directory can become large and unwieldy. You may want your own files kept separate from a co-worker's; or, you may want to organize your programs into categories that are convenient for you.

In an office you can separate files by putting them in different filing cabinets. MS-DOS allows you to separate files into directories. Directories are a convenient way of dividing your files into groups. For example, you may want all of your accounting programs in one directory, and text files in another. Any one directory can contain any number of files, and it may also contain other directories (referred to as subdirectories). This method of organizing your files is called a hierarchical directory structure.

A hierarchical directory structure can be thought of as a "tree." Just imagine the directories are branches of the tree and files are the leaves, except that the tree grows downward; that is, the "root" is at the top. The root is the first level in the directory structure. It is the directory that is automatically created when you format a diskette and fixed disk volume and start putting files on it. You can create additional directories and subdirectories by following the instructions in Chapter 4.

The tree or file structure grows as you create new directories for groups of files. Within each new directory, files can be added, or new subdirectories can be created.

It is possible for you to "travel" around this tree; for instance, it is possible to find any file in the system by starting at the root and traveling down any of the branches to the desired file. Conversely, you can start where you are within the file system and travel towards the root.

The filenames discussed earlier in this chapter are relative to your current directory and do not apply system-wide. For example, you have a diskette containing a number of files you want to work on in drive A. When you turn on your computer, you are "in" your directory. Unless you specifically create your file in a different directory, the new file is created in the directory in which you are now working. Users can have files of the same name that are unrelated because each is in a different directory.

Figure 3-1 illustrates a typical hierarchical directory structure.

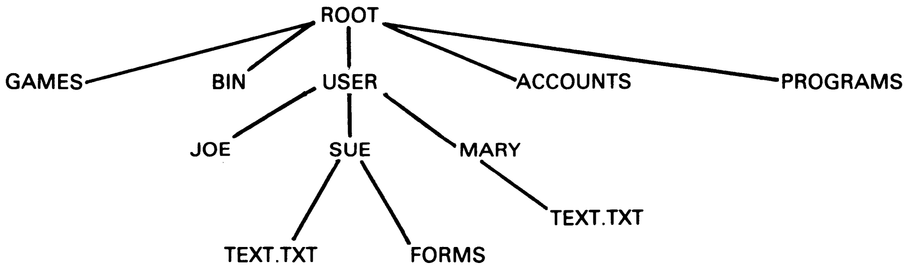


Figure 3-1 A Sample Hierarchical Directory Structure

The root directory (ROOT in Figure 3-1) is the first level in the directory structure. You can create subdirectories from the ROOT by using the MKDIR command (refer to Chapter 4 for information on MKDIR). In this example, five subdirectories of ROOT have been created:

- a directory of games, named GAMES
- a directory of all external commands, named BIN (refer to Chapter 4 for more information on the BIN directory)
- a directory called USER containing separate subdirectories for all users of the system
- a directory containing accounting information, named ACCOUNTS
- a directory of programs, named PROGRAMS

Joe, Sue, and Mary each have their own directories that are subdirectories of the USER directory. Sue has a subdirectory under the \USER\SUE directory named FORMS. Sue and Mary have files in their directories, each named TEXT.TXT. Notice that Mary's text file is unrelated to Sue's.

This organization of files and directories is not important if you work only with files in your own directory; but if you work with someone else or on several projects at one time, the hierarchical directory structure becomes quite useful. For example, you could get a list of the files in Sue's FORMS directory by entering

```
DIR \USER\SUE\FORMS
```

Note that the back slash (\) is used to separate directories from other directories and files.

To find out what files Mary has in her directory, you would enter

```
DIR \USER\MARY
```

FILENAMES AND PATHS

When you use hierarchical directories, you must tell MS-DOS where the files are located in the directory structure. Both Mary and Sue, for example, have files named TEXT.TXT. Each will have to tell MS-DOS in which directory her file resides when she wants to access it. This is done by giving MS-DOS a pathname to the file.

Pathnames

A simple filename is a sequence of characters that can optionally be preceded by a drive designation and followed by a file extension. A pathname is a sequence of directory names followed by a simple filename, each separated from the previous one by a back slash (\). The syntax of pathnames is

```
[<d>:] [<directory>] \ [<directory...>] \ [<filename>]
```

If a pathname begins with a back slash, MS-DOS searches for the file beginning at the root (or top) of the tree. Otherwise, MS-DOS begins at your current directory (known as the working directory) and searches downward from there. The pathname of Sue's TEXT.TXT file is \USER\SUE\TEXT.TXT.

When you are in your working directory, a filename and its corresponding pathname may be used interchangeably. Some sample names follow:

\	Indicates the root directory.
\PROGRAMS	Sample directory under the root directory containing program files.
\USER\MARY \FORMS\1A	A typical full pathname. This one happens to be a file named 1A in the directory named FORMS belonging to the user named Mary.
USER\SUE	A relative pathname; it names the file or directory SUE in subdirectory USER of the working directory. If the working directory is the root (\), it names \BIN\SUE.
TEXT.TXT	Name of a file or directory in the working directory.

MS-DOS provides special shorthand notations for the working directory and the parent directory (one level up) of the working directory

.	MS-DOS uses this shorthand notation to indicate the name of the working directory in all hierarchical directory listings. MS-DOS automatically creates this entry when a directory is made.
..	The shorthand name of the working directory's parent directory. If you enter

```
DIR..
```

then MS-DOS will list the files in the parent directory of your working directory.

If you enter

```
DIR..\..
```

then MS-DOS will list the files in the "parent's" parent directory.

Pathing and External Commands

As you know, external commands reside on storage media as program files. They must be read from the diskette or hard disk before they execute. (For more information on external commands, refer to Chapter 4.)

If you are working with more than one directory, it is suggested you put all MS-DOS external commands into a separate directory so they do not clutter your other directories. When you issue an external command to MS-DOS, it immediately checks your working directory to find that command. You must tell MS-DOS in which directory these external commands reside. This is done with the PATH command.

For example, if you are in a working directory named \BIN\PROG, and all MS-DOS external commands are in \BIN, you must tell MS-DOS to choose the \BIN path to find the FORMAT command. The command

```
PATH \BIN
```

tells MS-DOS to search in your working directory and the \BIN directory for all commands. If you want to know what the current path is, enter the word PATH; the current value of PATH will be printed.

For more information on the PATH command, refer to Chapter 4.

Pathing and Internal Commands

Internal commands are the simplest, most commonly used commands. They execute immediately because they are incorporated into the command processor. (For more information, refer to Chapter 4.)

Some internal commands can use paths. The four commands, COPY, DIR, DEL, and TYPE, have greater flexibility when you specify a pathname after the command. The syntax of these four commands is shown below.

COPY <pathname pathname>

If the second pathname to COPY is a directory, all files are copied into that directory. The first pathname may only specify files in the working directory.

DEL <pathname>

If the pathname is a directory, all the files in that directory are deleted. Note that the prompt "Are you sure (Y/N)?" will be displayed if you try to delete a path. Type Y to complete the command, or N to cancel the command.

DIR <pathname>

Displays the directory for a specific path.

TYPE <pathname>

You must specify a file in a path for this command. MS-DOS will display the file on your screen in response to the TYPE pathname command.

DISPLAYING YOUR WORKING DIRECTORY

All commands are executed while you are in your working directory. You can find out the name of the directory you are in by issuing the CHDIR (change directory) command with no options. For example, if your current directory is \USER \JOE, when you enter

```
CHDIR
```

you will see

```
A: \USER\JOE
```

This is your current drive designation plus the working directory (\USER\JOE).

If you now want to see what is in the \USER\ JOE directory, you can issue the DIR command. The following is an example of the display you might receive from the DIR command for a subdirectory on diskette.

```
     Volume in drive A has no ID
     Directory of A: \USER\JOE

.                     <DIR>                  8-09-83     10:09a
..                    <DIR>                  8-09-83     10:09a
TEXT                  <DIR>                  8-09-83     10:09a
FILE1       COM                    5243      8-04-83      9:30a
            4 File(s)              8376320 bytes free
```

A volume ID for this diskette was not assigned when it was formatted. Note that MS-DOS lists both files and directories in this output. As you can see, Joe has another directory in this tree structure named TEXT. The "." indicates the working directory \USER\JOE, and the ".." is the shorthand notation for the parent directory \USER. FILE1.COM is a file in the \USER\JOE directory. All of these directories and files reside on the diskette in drive A.

Because files and directories are listed together (see previous display), MS-DOS does not allow you to give a subdirectory the same name as a file in that directory. For example, if you have a path \BIN\USER\JOE where JOE is a subdirectory, you cannot create a file in the USER directory named JOE.

CREATING A DIRECTORY

To create a subdirectory in your working directory, use the MKDIR (make directory) command. For example, to create a new directory named NEWDIR under your working directory, simply enter

```
     MKDIR NEWDIR
```

After this command has been executed by MS-DOS, a new directory will exist in your tree structure under your working directory. You can also make directories anywhere in the tree structure by specifying MKDIR and then a pathname. MS-DOS will automatically create the . and .. entries in the new directory.

To put files in the new directory, use EDLIN (see Chapter 5).

HOW TO CHANGE YOUR WORKING DIRECTORY

Changing from your working directory to another directory is very easy in MS-DOS. Simply issue the CHDIR (Change Directory) command and supply a pathname. For example,

```
A:CHDIR \USER
```

changes the working directory from \USER\JOE to\USER. You can specify any pathname after the command to "travel" to different branches and leaves of the directory tree. The command "CHDIR.." will always put you in the parent directory of your working directory.

HOW TO REMOVE A DIRECTORY

To delete a directory in the tree structure, use the MS-DOS RMDIR (remove directory) command. For example, to remove the directory NEWDIR from the working directory, enter

```
RMDIR NEWDIR
```

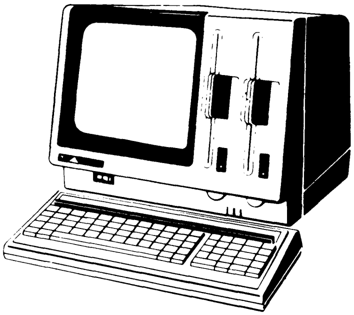
Note that the directory NEWDIR must be empty except for the . and .. entries before it can be removed; this will prevent you from accidentally deleting files and directories. You can remove any directory by specifying its pathname. To remove the \BIN\USER\JOE directory, make sure that it has only the . and .. entries, then enter

```
RMDIR \BIN\USER\JOE
```

To remove all the files in a directory (except for the . and .. entries), type DEL and then the pathname of the directory. For example, to delete all files in the \BIN\USER\SUE directory, enter

```
DEL \BIN\USER\SUE
```

You cannot delete the . and .. entries. They are created by MS-DOS as part of the hierarchical directory structure.



Chapter 4

MS-DOS Commands

The MS-DOS commands are what you use to instruct MS-DOS to perform tasks. MS-DOS commands...

- compare, copy, display, delete, and rename files
- copy and format disks
- execute system programs, such as EDLIN, as well as your own programs
- analyze and list directories
- enter the date, time, and remarks
- set various printer and screen options
- copy MS-DOS system files to another diskette.

You can enter commands in interactive mode by typing them individually at the keyboard. Or, you can "batch" commands in files (.BAT files) that execute on entry of the filenames as commands.

TYPES OF MS-DOS COMMANDS

There are two types of MS-DOS commands: internal and external.

Internal commands are the simplest, most commonly used commands. You cannot see these commands when you do a directory listing on your MS-DOS diskette; they are part of the command processor. When you type these commands, they execute immediately. Examples of internal commands that you already know are DIR and PATH.

External commands reside on diskettes or hard disks as program files. They must be read into memory from the storage device before they can execute.

Any filename with a filename extension of .COM, .EXE or .BAT is classified as an external command. For example, programs such as FORMAT.COM and COMP.COM are external commands. You can create your own commands and add them to the system. Programs that you develop with most languages (including assembly language) will be .EXE files. When you enter an external command, however, do not include its filename extension.

MS-DOS COMMAND FORMAT

MS-DOS commands consist of a command keyword and options. Options in your MS-DOS commands specify additional information to the system. If you do not include some options, MS-DOS provides a default value.

The following is the general format for all MS-DOS commands:

```
Command [options...]
```

where options represents one or more of the following items that provide additional directions:

d:	Refers to a diskette or hard disk drive designation.
filename	Refers to any valid name for a diskette or hard disk file, including an optional filename extension. The filename option does not refer to a device or to a drive designation.
.ext	Refers to an optional filename extension consisting of a period and one to three characters. When used, filename extensions immediately follow filenames.
filespec	Refers to an optional drive designation, a filename, and an optional three-letter filename extension in the following format: [<d:>] <filename> [<.ext>]
pathname	Refers to a pathname or filename in the following format: [<directory>] \ [<directory...>] [<filename>]
switches	Refers to options that select sub-functions of MS-DOS commands. They are preceded by a slash; for example, /P.
arguments	Refers to more information to MS-DOS commands. You usually choose between arguments; for example, ON or OFF.

INFORMATION COMMON TO ALL MS-DOS COMMANDS

The following information is common to all MS-DOS commands.

1. Commands are usually followed by one or more options.
2. Commands and options may be entered in uppercase, lowercase, or a combination of both.
3. Commands and their options must be separated by delimiters. Because they are easiest, you will use the space and comma as delimiters. For example,

   ```
   DEL MYFILE.OLD NEWFILE.TXT
   RENAME,THISFILE THATFILE
   ```

   You can also use the semicolon (;), the equal sign (=), or the TAB key as delimiters in MS-DOS commands.
4. Do not separate the parts of a file specification with other delimiters, since the colon and the period already serve as delimiters.
5. When instructions say "Press any key," you can press any alphabetic (A-Z) or numeric (0-9) key.
6. You must include the filename extension when referring to a file that already has a filename extension.
7. You can cancel commands when they are running by pressing CTRL-C.
8. Commands take effect only after you have pressed RETURN.
9. Wild cards (global filename characters) and file extensions for devices names (for example, PRN or CON) are not allowed in the names of any commands.
10. When commands produce more output than can be on the screen, the display will automatically scroll to the next screen. You can press CTRL-S to suspend scrolling. Press any key to resume the scrolling.
11. MS-DOS editing and function keys can be used when you enter commands. Refer to Chapter 4 for a complete description of these keys.
12. The prompt from the command processor is the default drive designation plus a greater than sign; for example, A>.
13. Diskette and hard disk drives will sometimes be referred to as source drives and destination drives. A source drive is the drive you will be transferring information from. A destination drive is the drive you will be transferring information to.

THE TEMPLATE

The template is a feature of MS-DOS that can simplify entering the same or similar commands one after the other.

The template is actually a buffer, or storage area, in memory where up to 127 characters of command information is temporarily stored. By using specific keys (see the discussions of the dual-mode function keys and control character functions), you can redisplay command information, change it, and reenter it.

MS-DOS operates as follows when you enter a command.

1. When you type information at the keyboard, it is held in the command line until you press RETURN.
2. Pressing RETURN causes the contents of the command line to be sent to the command processor for processing.
3. Pressing RETURN also copies the command line to the template.

The template always contains the last entered command line.

To retrieve the last entered command line and edit it, you use several of the dual-mode function (PF) keys at the top of your APC keyboard. The names given the keys for command line editing functions are shown on the paper "template" located above the PF keys. Table 4-1 lists the editing keys and describes their functions.

Table 4-1 Dual-Mode Function Keys Used in Command Line Editing

KEY NAME	PF KEY	EDITING FUNCTION
COPY1	PF1	Copies one character from the template to the command line
COPYUP	PF2	Copies characters up to the character specified in the template and puts these characters on the command line
COPYALL	PF3	Copies all remaining characters in the template to the command line

Table 4-1 Dual-Mode Function Keys Used in Command Line Editing (cont'd)

KEY NAME	PF KEY	EDITING FUNCTION
SKIP1	PF4	Skips over (does not copy) a character in the template
SKIPUP	PF5	Skips over (does not copy) the characters in the template up to the character specified
VOID	PF6	Voids the current input; leaves the template unchanged
INSERT	PF7	Enters insert mode
REPLACE	PF8	Turns insert mode off; this is the default mode
NEWLINE	PF9	Makes the new line the new template

For example, if you enter the following command:

```
DIR PROG.COM
```

MS-DOS displays information about the file PROG.COM on your screen. The command line is also saved in the template. To repeat the command, just press two keys: PF3 and RETURN.

The repeated command is displayed on the screen as you enter, as shown below:

```
DIR PROG.COM
```

Note again that pressing PF3 causes the contents of the template to be copied to the command line; pressing RETURN causes the command line to be sent to the command processor for execution.

If you want to display information about a file named PROG.ASM, you can use the contents of the template and enter

```
PF5 C
```

Pressing PF5 C copies all characters from the template to the command line, up to but not including C. MS-DOS displays

```
DIR PROG._
```

Now type

```
.ASM
```

The result is:

```
DIR PROG.ASM.
```

The command line DIR EDLIN.ASM is now in the template and ready to be sent to the command processor for execution. To do this, press RETURN.

Now assume that you want to execute the following command:

```
TYPE PROG.ASM
```

To do this, type

```
TYPE
```

and press PF7, space, PF3, RETURN.

When you are typing, the characters are entered directly into the command line where they overwrite corresponding characters in the template. Thus, the characters TYPE replace the characters DIR in the template. To insert a space between TYPE and PROG.ASM, you pressed PF7 and then SPACE. Note the automatic replacement feature is turned off when you press PF7. Finally, to copy the rest of the template to the command line, you pressed PF3 and then RETURN. The command TYPE PROG.ASM was processed by MS-DOS, and the template became "TYPE PROG.ASM."

If you had misspelled TYPE as BYTE, a command error would have occurred. Still, instead of throwing away the whole command, you could save the misspelled line before you pressed RETURN by creating a new template with the PF9 key.

```
BYTE PROG.ASM PF9
```

The PF9 key displays an @ after the last character displayed and moves to the first character position in a new command line. You could then edit this erroneous command by entering

```
T PF1 P PF3
```

Typing T replaces B with T. The PF1 key copies the next character from the template to the command line. The PF3 key copies the rest of the characters to the command line. The resulting command line is then the command that you want:

```
TYPE PROG.ASM
```

As an alternative, you can use the same template containing BYTE PROG.ASM and then use the PF4 and PF7 keys to achieve the same result:

```
PF4   PF4   PF1   PF7   YP   PF3
```

To illustrate how the command line is affected as you type, examine the keys typed on the left; their effect on the command line is shown on the right.

PF4	—	Skips over 1st template character
PF4	—	Skips over 2nd template character
PF1	T	Copies 3rd template character
PF7 YP	TYP	Inserts two characters
PF3	TYPE PROG.ASM	Copies rest of template

Note that pressing PF4 does not affect the displayed command line. It affects the template by deleting the first character. Similarly, the PF5 key deletes characters in the template, up to but not including a given character.

CONTROL CHARACTER FUNCTIONS

Certain combined keystrokes, CTRL and a letter, provide character control functions. A control character function is a function that affects the command line.

See Table 4-2 for a description of each function. This table also shows the single keys that perform the same functions when pressed.

Table 4-2 Control Character Functions

COMBINED KEYSTROKE	FUNCTION
CTRL-N	Cancels the echoing of keyboard input to line printer.
CTRL-C	Aborts the current command. The character C appears on the screen. Same as pressing PRINT.
CTRL-B	Removes the last character from the command line, and erases the character from the screen. Same as pressing BACK SPACE.
CTRL-J	Inserts a physical end-of-line, but does not empty the command line. Use the key to extend the current logical line beyond the physical limits of one screen.
CTRL-P	Echoes screen output to the line printer. Same as pressing PRINT.
CTRL-S	Suspends the display of output to the screen. Press any key to resume. Same as pressing BREAK STOP.
CTRL-X	Cancels the current line; by emptying the command line displaying a backslash (\), and effecting a carriage return and line feed. The template used by the editing commands is not affected. Same as pressing DEL.
CTRL-I	Moves the cursor to the next tab stop on the command line. Same as pressing TAB.

Remember that when you use a control character function, such as CTRL-C, you must hold down the CTRL key and then press the C key.

BATCH PROCESSING

Often you may find yourself typing the same sequence of commands again and again to perform a task. With MS-DOS, you can put the command sequence into a "batch file," and execute the entire sequence simply by typing the name of the batch file. "Batches" of commands in such files are processed as if they were typed at your keyboard. Each batch file must be named with the BAT file extension, and is executed by typing the filename without its extension.

You can create a batch file by using EDLIN or the COPY command. Refer to the section on creating an AUTOEXEC.BAT file for more information.

Several commands are available for use in batch files: REM, PAUSE, ECHO, FOR, GOTO, IF, and SHIFT. REM permits you to include remarks in your batch files without these remarks being executed as commands. PAUSE prompts you with an optional message and permits you either to continue or end the batch process at a given point. ECHO displays the commands in a batch file on the screen as they are executed. FOR repeats an operation for the number of times it encounters a specified input value. GOTO performs a command that follows a label for an infinite number of times. IF tests for a condition to be met by processing, then executes another command if the condition tests true. SHIFT overrides the 10-parameter limit for a command line by removing the first parameter value in successive shifts so that additional parameters can be entered. Some of these commands can be used in interactive processing as well. See the descriptions of the individual commands for more information.

Batch processing is useful if you want to execute several MS-DOS commands with one batch command, such as when you format and check a new disk. For example, a batch file for this purpose might look like this:

```
1: REM This is a file to check new disks
2: REM It is named NEWDISK.BAT
3: PAUSE Insert new disk in drive B:
4: FORMAT B:
5: DIR B:
6: CHKDSK B:
```

To execute this .BAT file, simply enter the filename without the .BAT extension.

```
NEWDISK
```

There are two steps to using batch files.

- First you create a batch file with a file extension of BAT in one of your directories on diskette or hard disk.
- Then you execute the batch file by entering its batch filename (without the BAT extension) and device designation, if required as a command.

Batch file processing should begin at this point.

Before proceeding, however, read the following information about batch processing.

1. Do not enter the filename BATCH (unless the name of the file you want to execute is BATCH.BAT).
2. Only the filename should be entered to execute the batch file. Do not enter the filename extension.
3. The commands in the file named <filename>.BAT are executed.
4. If you press CTRL-C while in batch mode, this prompt appears:

   Terminate batch job (Y/N)?

   If you press Y, the remainder of the commands in the batch file are ignored and the system prompt appears.

   If you press N, only the current command ends and batch processing continues with the next command in the file.
5. If you remove the diskette containing a batch file being executed, MS-DOS prompts you to insert it again before the next command can be read.
6. The last command in a batch file may be the name of another batch file. This allows you to call one batch file from another when the first is finished.

THE AUTOEXEC.BAT FILE

As discussed in Chapter 2, an AUTOEXEC.BAT file allows you to automatically execute programs when you start MS-DOS. Automatic program execution is useful when you want to execute a batch program automatically each time you start the system and be able to run other programs as well.

The AUTOEXEC.BAT file is a batch file that is automatically executed each time you start the system. When you start MS-DOS, the command processor searches the MS-DOS diskette for this file.

If MS-DOS finds it, the AUTOEXEC.BAT file is immediately executed by the command processor and the date and time prompts are bypassed.

If MS-DOS does not find an AUTOEXEC.BAT file the date and time prompts are issued. Figure 4-1 illustrates the automatic program execution operation.

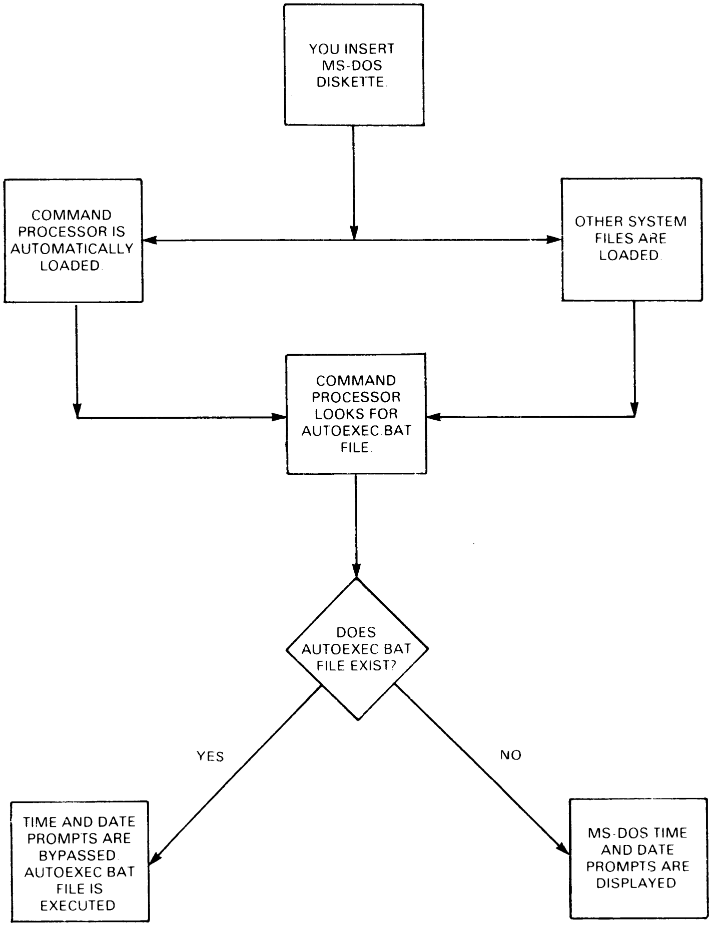

YOU INSERT MS-DOS DISKETTE.
COMMAND PROCESSOR IS AUTOMATICALLY LOADED.
OTHER SYSTEM FILES ARE LOADED.
COMMAND PROCESSOR LOOKS FOR AUTOEXEC.BAT FILE.
DOES AUTOEXEC.BAT FILE EXIST?
YES
NO
TIME AND DATE PROMPTS ARE BYPASSED AUTOEXEC.BAT FILE IS EXECUTED
MS-DOS TIME AND DATE PROMPTS ARE DISPLAYED

Figure 4-1 How MS-DOS Uses the AUTOEXEC.BAT File

How to Create an AUTOEXEC.BAT File

If, for example, you wanted to automatically load BASIC and run a program called MENU each time you started MS-DOS, you could create an AUTOEXEC.BAT file as follows:

directory name [illegible]
on prompt
Pg

1. Enter

   `COPY CON: AUTOEXEC.BAT`

   This statement tells MS-DOS to copy the information from the keyboard (console) into the AUTOEXEC.BAT file. Note that the AUTOEXEC.BAT file must be created in the root directory of your MS-DOS diskette.
2. Now enter

   `BASIC MENU`

   This statement goes into the AUTOEXEC.BAT file. It tells MS-DOS to load BASIC and run the MENU program whenever MS-DOS is started.
3. Press CTRL-Z; then press RETURN to put the command BASIC MENU in the AUTOEXEC.BAT file.
4. The MENU program will now run automatically whenever you start MS-DOS.

You can enter any MS-DOS command or series of commands in the AUTOEXEC.BAT file that you create.

NOTE

> Remember that if you use an AUTOEXEC.BAT file, MS-DOS will not prompt you for a current date and time unless you include the DATE and TIME commands in the AUTOEXEC.BAT file. It is strongly recommended that you include these two commands in your AUTOEXEC.BAT file, since MS-DOS uses this information to keep your directory current.

Creating a .BAT File with Replaceable Parameters

There may be times when you want to create an application program and run it with different sets of data. These data may be stored in various MS-DOS files.

When used in MS-DOS commands, a parameter is an option that designates a replaceable value. With MS-DOS, you can create a batch (.BAT) file with dummy (replaceable) parameters. These parameters, named %0-%9, can be replaced by values supplied when the batch file executes.

For example, when you type the command line

```
COPY CON MYFILE.BAT
```

the next lines you type are copied from the console to a file named MYFILE.BAT on the default drive.

```
A:COPY CON MYFILE.BAT
COPY %1.MAC %2.MAC
TYPE %2.PRN
TYPE %0.BAT
```

Now, press CTRL-Z and then press RETURN. MS-DOS responds with this message:

```
1 File(s) copied
A_
```

The file MYFILE.BAT, which consists of three commands, now resides on the diskette in the default drive.

The dummy parameters %1 and %2 are replaced sequentially by the parameters you supply when you execute the file. The dummy parameter %0 is always replaced by the drive designator, if specified, and the filename of the batch file (for example, MYFILE).

Note the following rules for dummy parameter entry:

- Up to 10 dummy parameters (%0-%9) can be specified. Refer to the SHIFT command in Chapter 4 if you wish to specify more than 10 parameters.
- If you use the percent sign as part of a filename within a batch file, you must type it twice. For example, to specify the file ABC%.EXE, you must type it as ABC%%.EXE in the batch file.

Executing a .BAT File

To execute the batch file MYFILE.BAT and to specify the parameters that will replace the dummy parameters, you must enter the batch filename (without its extension) followed by the parameters you want MS-DOS to substitute for %1, %2, and so forth.

Remember that the file MYFILE.BAT consists of 3 lines:

```
COPY %1.MAC %2.MAC
TYPE %2.PRN
TYPE %0.BAT
```

To execute the MYFILE batch process, enter

```
MYFILE A:PROG1 B:PROG2
```

MYFILE is substituted for %0, A:PROG1 for %1, and B:PROG2 for %2.

The result is the same as if you had typed each of the commands in MYFILE with their parameters, as follows:

```
COPY A:PROG1.MAC B:PROG2.MAC
TYPE B:PROG2.PRN
TYPE MYFILE.BAT
```

The following illustration represents how MS-DOS replaces each of the above parameters.

BATCH FILENAME	PARAMETER1 (%0) (MYFILE)	PARAMETER2 (%1) (PROG1)	PARAMETER3 (%2) (PROG2)
MYFILE	MYFILE.BAT	PROG1.MAC	PROG2.MAC PROG2.PRN

Remember that the dummy parameter %0 is always replaced by the drive designator (if specified) and the filename of the batch file.

HOW MS-DOS CONTROLS INPUT AND OUTPUT

MS-DOS always assumes that input comes from the keyboard and output goes to the APC screen. However, the flow of command input and output can be redirected. Input can come from a file rather than the keyboard, and output can go to a file or to a line printer instead of to the screen. In addition, "pipes" can be created that allow output from one command to become the input to another. Redirection and pipes are discussed in the next sections.

Redirecting Output

Most commands produce output that is sent to the APC screen. You can send this information to a file by using a greater than sign (>) in your command. For example, the command

```
DIR
```

displays a directory listing of the diskette or hard disk volume in the default drive on the screen. The same command can send this output to a file named MYFILES by designating the output file on the command line:

```
DIR >MYFILES
```

If the file MYFILES does not already exist, MS-DOS creates it and stores your directory listing in it. If MYFILES already exists, MS-DOS overwrites what is in the file with the new data.

If you want to append your directory or a file to another file (instead of replacing the entire file), two greater than signs (> >) can be used to tell MS-DOS to append the output of the command (such as a directory listing) to the end of a specified file. The command

```
DIR >>MYFILES
```

appends your directory listing to a currently existing file named MYFILES. If MYFILES does not exist, it is created.

If is often useful to have input for a command come from a file rather than from the keyboard. This is possible to do by using a less than sign (<) in your command. For example, the command

```
SORT <NAMES> LIST1
```

sorts the file NAMES and sends the sorted output to a file named LIST1.

Filters

A filter is a command that reads your input, transforms it in some way, and then sends it out, usually to your screen or to a file. Data treated in this manner is said to have been "filtered" by the program. Since filters can be put together in many different ways, a few filters can take the place of a large number of commands.

MS-DOS filters include the CIPHER, FIND, MORE, and SORT commands. Their functions can be briefly described as follows:

CIPHER — Encrypts/decrypts a file.

FIND — Searches for a constant string of text in a file.

MORE — Takes standard console output and displays it, one screen at a time.

SORT — Sorts text.

You can see how these filters are used in the next section.

Command Piping

If you want to give more than one command to the system at a time, you can "pipe" commands to MS-DOS. For example, you may occasionally need to have the output of one program sent as the input to another program. A typical case would be a program that produces output in columns. It could be desirable to have this columnar output sorted.

Piping is done by separating commands with the pipe separator, which is the vertical bar symbol (|). For example, the command

```
DIR | SORT
```

will give you an alphabetically sorted listing of your directory. The vertical bar causes all output generated by the command on the left side of the bar to be sent to the program on the right side of the bar for processing.

Piping can also be used when you want to send output to a file. If you wanted your directory sorted and sent to a new file (for example, DIREC.FIL), you would enter

```
DIR | SORT >DIREC.FIL
```

MS-DOS will create a file named DIREC.FIL on your default drive. DIREC.FIL a sorted listing of the directory on the default drive, since no other drive was specified in the command. To specify a drive other than the default drive, enter

```
DIR| SORT B:DIREC.FIL
```

This sends the sorted data to a file named DIRPC.FIL on drive B.

A pipeline may consist of more than two commands.

For example,

```
DIR| SORT| MORE
```

will sort your directory, show it to you one screen at a time, and put the message "--MORE--" at the bottom of your screen as long as there is more output to be seen.

You will find many uses for piping commands and filters.

DISKETTE AND HARD DISK ERRORS

If an error occurs any time that a command is trying to access a diskette or hard disk drive, MS-DOS retries the operation three times. If the operation cannot be completed successfully, MS-DOS returns an error message in the following format:

```
<yyy> ERROR WHILE I/O action ON DRIVE x
Abort,Ignore,Retry:
```

In this message, <yyy> may be one of the following:

WRITE PROTECT
NOT READY
SEEK
DATA
SECTOR NOT FOUND
WRITE FAULT
DISKETTE
DISK

The I/O action may be either of the following:

READING
WRITING

The drive x indicates the drive in which the error has occurred.

MS-DOS waits for you to enter one of the following responses:

A	Abort. Terminate the program requesting the disk read or write.
I	Ignore. Ignore the error, as if it did not occur.
R	Retry. Repeat the operation. Type R when you have corrected the error, for instance, by re-inserting a diskette when the message is "NOT READY," or removing a write-protect tab when the message is "WRITE PROTECT."

Usually, you will want to attempt recovery by entering responses in this order:

R	(to try again)
A	(to terminate the program and try a new device)

One other error message might be related to a faulty diskette or hard disk read or write:

```
FILE ALLOCATION TABLE BAD FOR DRIVE x
```

This message means that the copy in memory of one of the file allocation tables has pointers to nonexistent blocks. Possibly a diskette or hard disk was incorrectly formatted or not formatted before use. If this error persists, the storage medium is currently unusable and must be formatted prior to use. Unfortunately, data and programs on it might be lost.

DUAL-MODE FUNCTION KEYS USED FOR COMMAND ENTRY

Several of the dual-mode function (PF) keys at the top of the APC keyboard have been pre-programmed to provide access to commonly used commands. Using the SHIFT key and PF keys 1 through 16, you can simplify command entry for the system functions or programs whose names appear on the template above the keys. Table 4-3 lists the commands, their associated PF keys, and describes their functions with the responses required from you, if any.

BREAK Command

Syntax:

```
BREAK ON|OFF
```

Type:

Internal

Function:

Sets the CTRL-C check.

Comments:

If you are running an application program that uses CTRL-C, you will want to turn off the MS-DOS CTRL-C function so that when you press CTRL-C you affect your program and not the operating system. Specify BREAK OFF to turn off CTRL-C and BREAK ON when you have finished running your application program.

CAT Command

Syntax:

CAT [filespec]

Type:

External

Function:

Catalogs a directory.

Comments:

This command sorts and formats a specified directory and displays it; for example,

```
B:CAT

CHKDSK   .COM  6k  FORMAT   .COM  5k  PRINT    .COM  3k
DEBUG    .COM 10k  LINK     .EXE 41k  TRACKS    <DIR>
EDLIN    .COM  4k  LOCATE   .EXE  2k  T96      .COM  3k

Total of 75 bytes in 9 files, 673 bytes remain available.
```

If you type a file specification after CAT, this command will display directory information about the file.

CHDIR Command

Syntax:

```
CHDIR [pathname]
```

Synonym:

CD

Type:

Internal

Function:

Changes a directory to a different path; displays current (working) directory.

Comments:

If your working directory is \BIN\USER\JOE and you want to change your path to another directory (such as \BIN\USER\JOE\FORMS), enter

```
CHDIR \BIN\USER\JOE\FORMS
```

and MS-DOS will put you in the new directory. A shorthand notation is also available with this command:

```
CHDIR..
```

This command will always put you in the parent directory of your working directory.

CHDIR used without a pathname displays your working directory. If your working directory is \BIN\USER\JOE on drive B, and you enter CHDIR, MS-DOS will display

```
B: \BIN\USER\JOE
```

This command is useful if you forget the name of your working directory.

CHKDSK Command

Syntax:

```
CHKDSK [d:] <filespec> [/F] [/V]
```

Type:

External

Function:

Scans a directory on the specified diskette or hard disk drive and checks it for consistency.

Comments:

CHKDSK should be run occasionally on each diskette and hard disk to check for errors in directories. If any errors are found, CHKDSK will display error messages, if any, and then a status report.

A sample status report for a diskette directory follows:

```
160256   bytes total diskette space
  8192   bytes in 2 hidden files
   512   bytes in 2 directories
 30720   bytes in 8 user files
121344   bytes available on diskette
 65536   bytes total memory
 53152   bytes free
```

CHKDSK will automatically attempt to correct several of the errors found in your directory if you specify the /F (fix) switch. However, it will not correct others, even if you have specified /F. See Table 4-4 for a list of CHKDSK errors and the procedures to correct them.

Note that typing /V causes CHKDSK to display the names of each subdirectory and individual file on the specified diskette or hard disk volume. This is a convenient way to display or print a list of all the subdirectories and files on either type of volume.

You can redirect the output from CHKDSK to a file. Simply enter

```
CHKDSK A:>filename
```

The errors will be sent to the filename specified. Do not use the /F switch if you redirect CHKDSK output.

Table 4-4 CHKDSK Error Messages

ERROR MESSAGE	CORRECTIVE PROCEDURE
Invalid drive specification	Reenter the drive specification.
Invalid parameter	Reenter the appropriate parameter, or option.
Invalid subdirectory entry	Reenter the subdirectory name.
Cannot CHDIR to filename Tree past this point not processed	
First cluster number is invalid entry truncated	
Allocation error, size adjusted	
Disk error reading FAT (file allocation table)	
Disk error writing FAT (file allocation table)	
<filename> contains non-contiguous blocks	
All specified file(s) are contiguous	
Errors found, F parameter not specified Corrections will not be written to disk.	You must specify the /F switch if you want the errors corrected by CHKDSK.
Incorrect DOS version	You cannot run CHKDSK on versions of MS-DOS that are not 2.0 or higher.
Insufficient memory Processing cannot continue	There is not enough memory in your APC to process CHKDSK for the specified device. You must obtain more memory to run CHKDSK.
Invalid current directory Processing cannot continue	Restart the system and rerun CHKDSK.
Cannot CHDIR to root Processing cannot continue	The device you are checking is bad. Try restarting MS-DOS and RECOVER the disk.

Table 4-4 CHKDSK Error Messages (cont'd)

ERROR MESSAGE	CORRECTIVE PROCEDURE
<filename> is cross-linked on cluster	Make a copy of the file you want to keep, and then delete both files that are cross linked.
X lost clusters found in y chains Convert lost chains to files (Y/N)?	If you respond Y to this prompt, CHKDSK will create a directory entry and a file for you to resolve this problem (files created by CHKDSK are named FILEnnnnnnn). CHKDSK will then display X BYTES DISK SPACE FREED If you respond N to this prompt and have not specified the /F switch, CHKDSK frees the clusters and displays X BYTES DISK SPACE WOULD BE FREED
Probable non-DOS diskette Continue (Y/N)?	The diskette you are using is a non-MS-DOS diskette. You must indicate whether or not you want CHKDSK to continue processing.
Insufficient room in root directory Erase files in root and repeat CHKDSK	CHKDSK cannot process until you delete files in the root directory.
Unrecoverable error in directory Convert directory to file (Y/N)?	If you respond Y to this prompt, CHKDSK will convert the bad directory into a file. You can then fix the directory yourself or delete it.

CHR Command

Syntax:

CHR

Type:

External

Function:

Generates and maintains an auxiliary character set.

Comments:

CHR creates, changes, or loads into memory an auxiliary character set stored in a data file with the file extension CHR. Any number of auxiliary character sets can be created, each containing up to 256 character patterns identified by the hexadecimal values 00 to FF. These auxiliary character set files are then available for use by application programs. Each auxiliary character within a set is constructed within an 8 by 16 matrix using the commands described below.

CHR has two functions: 1) loading an existing character set from diskette or hard disk to memory, and 2) updating or creating the contents of a character set. After you enter the command line as shown above, the following prompt appears

```
UPDATE OR LOAD (U,L)?
```

If you enter L, you must then identify the input filename in one of the following formats:

- <filename>
- <d:filename>

If you only press RETURN, CHR automatically searches the default device for the filename AUXCG.CHR, which is the default auxiliary character file. If the input file is located, the auxiliary character set is loaded into memory starting at hexadecimal address 0D8000.

If you enter U to update an auxiliary character file, the filename you enter for the input file determines whether you will be maintaining an existing file or creating a new one. If the specified filename does not exist, CHR creates a file on the d: drive using the name you entered.

w/ ALT key engaged all char displayed [illegible] currently loaded alternate char set.

An output file specification must then be entered. The output file will store the changed version of the character set. The entry format is the same as for the input file specification. If you only press RETURN the filename of the output file is the same as that for the input file.

When you press certain keys during CHR's operations you issue sub-commands. Table 4-5 lists the keys you can use and the functions they perform.

Table 4-5 CHR Sub-Commands

SUB-COMMAND	KEY USED	FUNCTION
CURSOR UP	↑	Moves the cursor upward one row at a time within the matrix.
CURSOR DOWN	↓	Moves the cursor downward one row at a time within the matrix.
CURSOR LEFT	+	Moves the cursor to the left one column at a time within the matrix.
CURSOR RIGHT	-	Moves the cursor to the right one column at a time within the matrix.
BIT OFF	SPACE	Turns off the bit or graphic block at the current cursor position by overtyping with a space. The cursor then moves forward one position.
BIT ON	*	Turns on the bit at the current cursor position. The graphic block at the position appears highlighted, and the cursor moves forward one position.
CURSOR HOME	CLEAR HOME	Moves the cursor to the home position, the upper left corner of the matrix (0,0).
DISPLAY	D	Displays or redisplays the current character following modifications.
NEXT	RETURN	Displays the next character in the auxiliary character set according to the next hexadecimal code in sequence. Pressing RETURN at the last character in the file (code=FF) redisplays the first character (code=00).

Table 4-5 CHR Sub-Commands (cont'd)

SUB-COMMAND	KEY USED	FUNCTION
BACK	B	Displays the previous character in the auxiliary character set according to the previous hexadecimal code. Pressing B at the first character in the file (code=00) redisplays the last character (code=FF).
LOAD	L	Loads all characters of the auxiliary character set being updated or created into memory.
SEARCH CODE	C	Prompts for input of a hexadecimal code, then displays the character corresponding to that code. Pressing RETURN instead of entering a code redisplays the last character shown on the screen.
END	E	Displays a prompt confirming the end of auxiliary character set updating. Enter Y to end the program. When you are creating a new file, a prompt asking "CREATE filename?" appears. When you are updating an existing file, a prompt asking "UPDATE file name?" appears.

CIPHER Command

Syntax:

```
CIPHER <keyword> [<filename>]
```

Type:

External

Function:

Encrypts and decrypts files based on a specified keyword.

Comments:

Use this command when you want to encrypt a file for security purposes. The CIPHER command uses a keyword to encrypt the file. This keyword also acts as a password to restrict unauthorized access to the file.

To encrypt the file NSA.CIA using the keyword "SECRET", enter

```
CIPHER SECRET NSA.CIA
```

This will display the encrypted file (NSA.CIA) on your screen. If you want the encrypted file sent to another file, enter

```
CIPHER <NSA.CIA >MYSTERY.NEW
```

where MYSTERY.NEW is the name of the file where you are storing the encrypted file. You may delete the original file NSA.CIA.

To decrypt the encrypted file MYSTERY.NEW, simply reverse the process.

```
CIPHER SECRET <MYSTERY.NEW
```

This command decrypts the file MYSTERY.NEW and displays it on your screen. If you want the decrypted file sent to another file, called NOSECRET.XXX, type

```
CIPHER SECRET <MYSTERY.NEW >NOSECRET.XXX
```

NOTE

You must supply the same keyword that you encrypted the file with when you decrypt the file, or the CIPHER command will not work.

CLS Command

Syntax:

CLS

Type:

Internal

Function:

Clears the APC screen.

Comments:

The CLS command causes MS-DOS to send the ANSI escape sequence ESC[2J, which clears your screen, to your console.

COPY Command

Syntax:

```
COPY <filespec> [filespec] [pathname] [pathname] [/V]
```

copy \compu\filename (if you're in root, \, will copy filename

Type:

Internal or copy \compu\filename [illegible] with same [illegible] directory [illegible]

Function:

Copies one or more files from one diskette or hard disk drive to another. If you prefer, you can give the copies different names. This command can also copy files on the same device.

Comments:

If the second filespec option is not given, the copy will be on the default drive and will have the same name as the original file (first filespec option), thus creating a duplicate. If the first filespec is on the default drive and the second filespec is not specified, COPY will be canceled. Copying files to themselves is not allowed; MS-DOS will return the error message:

```
File cannot be copied onto itself
0 File(s) copied
```

The second option may take three forms:

1. If the second option is a drive designation (d:) only, the original file is copied with the original filename to the designated drive.
2. If the second option is a filename only, the original file is copied to a file on the default drive with the specified filename.
3. If the second option is a full filespec, the original file is copied to a file on the default drive with the specified filename.

The /V switch causes MS-DOS to verify that the sectors written on the destination drive are recorded properly. Although there are rarely recording errors when you run COPY, you can verify that critical data has been correctly recorded. This option causes the COPY command to run more slowly because MS-DOS must check each entry recorded on the storage medium.

The COPY command also allows file concatenation (joining) while copying. Concatenation is accomplished by simply listing any number of files as options to COPY, separated by +'s.

Example:

```
COPY A.XYZ + B.COM + B:C.TXT BIGFILE.CRP
```

This command concatenates files named A.XYZ, B.COM, and B:C.TXT and places them in the file on the default drive called BIGFILE.CRP.

To combine several files using wild cards into one file, you could type

```
COPY *.LST COMBIN.PRN
```

This command would take all files with a filename extension of .LST and combine them into a file named COMBIN.PRN.

In the following example, for each file found matching *.LST, that file is combined with the corresponding .REF file. The result is a file with the same filename but with the extension PRN. Thus, FILE1.LST will be combined with FILE1.REF to form FILE1.PRN; then XYZ.LST with XYZ.REF to form XYZ.PRN; and so on.

```
COPY *.LST + *.REF *.PRN
```

The following COPY command combines all files matching *.LST, then all files matching *.REF, into one file named COMBIN.PRN.

```
COPY *.LST + *.REF COMBIN.PRN
```

Do not enter a concatenation COPY command where one of the source filenames has the same extension as the destination. For example, the following command is an error if ALL.LST already exists.

```
COPY *.LST ALL.LST
```

The error would not be detected, however, until ALL.LST is appended. At this point it could have already been destroyed.

COPY compares the filename of the input file with the destination filename. If they are the same, that one input file is skipped, and the error message "Content of destination lost before copy" is printed. Further concatenation proceeds normally. This allows "summing" files, as in this example:

```
COPY ALL.LST + *.LST
```

This command appends all *.LST files, except ALL.LST itself, to ALL.LST. This command will not produce an error message and is the correct way to append files using the COPY command.

CTTY Command

Syntax:

```
CTTY \DEV
```

Type:

Internal

Function:

Allows you to change the device from which you issue commands (TTY represents the console).

Comments:

DEV stands for "device," i.e., the device from which you are giving commands to MS-DOS. This command is useful if you want to change the device on which you are working. The command

```
CTTY \AUX
```

moves all command I/O from the current device (the console) to the AUX port, such as another console used in data communications.

The command

```
CTTY \CON
```

moves I/O back to the console. Refer to the section on name files in Chapter 3, for a list of valid device names to use with the CTTY command.

DATE Command

Syntax:

```
DATE [<mm>-<dd>-<yy>]
```

Type:

Internal

Function:

Enters or changes the date known to the system. This date will be recorded in the directory for any files you create or alter.

You can change the date from your keyboard or from a batch file. (MS-DOS does not display a prompt for the date if you use an AUTOEXEC.BAT file, so you may want to include a DATE command in that file.)

Comments:

If you type DATE, the program will respond with the message:

```
Current date is <mm>-<dd>-<yy>
Enter new date:_
```

Press RETURN if you do not want to change the date shown.

You can also type a particular date after the DATE command, as in

```
DATE 3-9-81
```

In this case, you do not have to answer the "Enter new date:" prompt.

The new date must be entered using numerals only; letters are not permitted. The allowed options are

<mm> = 1-12
<dd> = 1-31
<yy> = 80-99 or 1980-2099

The date, month, and year entries may be separated by hyphens (-) or slashes (/). MS-DOS is programmed to change months and years correctly, whether the month has 31, 30, 29, or 28 days. MS-DOS handles leap years, too.

If the options or separators are not valid, DATE displays the message:

```
Invalid date
Enter new date:-
```

DATE then waits for you to enter a valid date. The entered date is displayed in the APC status line on your screen.

DEL Command

Syntax:

DEL [filespec][pathname]

Synonym:

ERASE

Type:

Internal

Function:

Deletes all files with the designated filespec.

Comments:

If the filespec is *.*, the prompt "Are you sure?" appears. If a "Y" or "y" is typed as a response, then all files are deleted as requested. You can also type ERASE for the DELETE command.

DIR Command

Syntax:

```
DIR [filespec][pathname][/P][/W]
```

Type:

Internal

Function:

Lists the files in a directory.

Comments:

If you just type DIR, all directory entries on the default drive are listed. If only the drive specification is given (DIR d:), all entries on the diskette or hard disk in the specified drive are listed. If only a filename is entered with no extension (DIR filename), then all files with the designated filename on the disk in the default drive are listed. If you designate a file specification (for example, DIR d:filename.ext), all files with the specified filename on the specified drive are listed. In all cases, files are listed with their size in bytes and with the time and date of their last modification.

The wild card characters ? and * (question mark and asterisk) may be used in the filename option. Note that the following DIR commands are equivalent:

COMMAND	EQUIVALENT
DIR	DIR *.*
DIR FILENAME	DIR FILENAME.*
DIR .EXT	DIR *.EXT
DIR .	DIR *.

Two switches may be specified with DIR. The /P switch selects Page Mode. With /P, display of the directory pauses after the screen is filled. To resume display of output, press any key.

The /W switch selects Wide Display. With /W, only filenames are displayed, without other file information. Files are displayed five per line.

DISKCOPY Command

Syntax:

DISKCOPY [d:] [d:]

Type:

External

Function:

Copies the contents of the diskette in the source drive to the diskette in the destination drive.

Comments:

The first option you specify is the source drive. The second option is the destination drive.

The diskette in the destination drive must be formatted prior to using DISKCOPY, using FORMAT.

You can specify the same drives or you may specify different drives. If the drives designated are the same, a single-drive copy operation is performed. (This operation is automatic in a single-drive system.) You are prompted to insert diskettes at the appropriate times. DISKCOPY waits for you to press any key before continuing.

After copying, DISKCOPY prompts:

```
Copy another (Y/N)?
```

If you press Y, the next copy is performed on the same drives that you originally specified, after you have been prompted to insert the proper diskettes.

To end DISKCOPY, press N.

Notes:

1. If you omit both options, a single-drive copy operation will be performed on the default drive.
2. If you omit the second option, the default drive will be used as the destination drive.
3. Both diskettes must have the same number of physical sectors and those sectors must be the same size.
4. Diskettes that have had a lot of file creation and deletion activity may become fragmented, because diskette space is not allocated sequentially. The first free sector found is the next sector allocated, regardless of its location on the diskette.

   A fragmented diskette can cause poor performance due to delays involved in finding, reading, or writing a file. If this is the case, you must use the COPY command, instead of DISKCOPY, to copy your diskette and eliminate the fragmentation.

   For example,

   ```
   COPY A:*.* B:
   ```

   copies all files from the diskette in drive A to the diskette in drive B.
5. DISKCOPY automatically determines the number of sides to copy, based on system information about the source drive and diskette.
6. If diskette errors are encountered during a DISKCOPY, MS-DOS displays

   ```
   DISK error while reading drive A
   Abort, Ignore, Retry?
   ```

   Refer to the section on diskette errors at the beginning of this chapter, if you need to be reminded about how to respond to this message.

DISKCOPY does not copy the contents of a diskette to a hard disk volume. You must use COPY to do this.

DISP Command

Syntax:

DISP

Type:

External

Function:

Displays the auxiliary character set that you load into memory.

Comments:

The character set is loaded using the CHR command. When you invoke DISP, the characters for all 256 hexadecimal codes in the currently loaded set are displayed in a matrix on the screen. Codes without characters assigned appear as blanks or miscellaneous images. If no character set is currently loaded, all positions appear as miscellaneous images.

DISP1 Command

Syntax:

DISP1

Type:

External

Function:

Displays the default character set for your system.

Comments:

The default character set for your system is stored in read only memory (ROM). This character set is the one that MS-DOS uses to display information on the APC screen unless you load an auxiliary character set (see the CHR command).

ECHO Command

Syntax:

ECHO [ON | OFF | message]

Type:

Internal

Function:

Turns batch echo feature on and off.

Comments:

Normally, commands in a batch file are displayed "echoed," or displayed on the APC screen when they are seen by the command processor. ECHO OFF turns off this feature. ECHO ON turns the echo back on.

If you type ECHO but do not specify ON or OFF, the current setting is displayed.

EXIT Command

Syntax:

```
EXIT
```

Type:

Internal

Function:

Exits the program COMMAND.COM (the command processor) and returns to a previous directory or command level, if one exists.

Comments:

This command can be used when you are running an application program and want to start the MS-DOS command processor, then return to your program. For example, to look at a directory on drive B while running an application program, you must start the command processor by typing COMMAND in response to the default drive prompt,

```
A:COMMAND
```

You can now type the DIR command and MS-DOS will display the directory for the default drive. When you type EXIT, you return to the previous level your application program.

FIND Command

Syntax:

```
FIND [/V /C /N] <string> [<filename...>]
```

Type:

External

Function:

Searches for a specific string of text in a file or files.

Comments:

FIND is a filter that takes as options a string and a series of filenames. It will display all lines that contain a specified string from the files specified in the command line.

If no files are specified, FIND will take the input on the screen and display all lines that contain the specified string.

Switches for FIND are

/V	that causes FIND to display all lines not containing the specified string.
/C	that causes FIND to print only the count of lines that contained a match in each of the files.
/N	that causes each line to be preceded by its relative line number in the file.

The string should be enclosed in quotes. For example,

```
FIND "Fool's Paradise" BOOK1.TXT BOOK2.TXT
```

displays all lines from BOOK1.TXT and BOOK2.TXT (in that order) that contain the string "Fool's Paradise." The command

```
DIR B: | FIND /V "DAT"
```

causes MS-DOS to display all names of the files on the disk in drive B which do not contain the string DAT. Type double quotes around a string that already has quotes in it.

When an error is detected, FIND responds with one of the error messages listed in Table 4-6.

Table 4-6 FIND Error Messages

ERROR	CORRECTIVE PROCEDURE
Incorrect DOS version	FIND will only run on versions of MS-DOS that are 2.0 or higher.
FIND: Invalid number of parameters	You did not specify a string when issuing the FIND command.
FIND: Syntax error	You typed an illegal string when issuing the FIND command.
FIND: File not found filename	The filename you have specified does not exist or FIND cannot find it.
FIND: Read error in filename	An error occurred when FIND tried to read the file specified in the command.
FIND: Invalid parameter option-name	You specified an option that does not exist.

FOR Command

Syntax:

```
FOR %%<c> IN set DO <command> — for batch processing
FOR %<c> IN set DO <command> — interactive processing
```

Type:

Internal

Function:

Does repetitive operations in batch and interactive file processing.

Comments:

The <command> that follows DO will be executed once for each occurrence of <c> in <set>.

<c> can be any character except 0,1,2,3,..,9 to avoid confusing values entered for interactive processes with the %0-%9 batch parameter values.

<set> is (#<item>*#)

The %%<c> variable is set sequentially to each member of <set> and then <command> is evaluated. If a member of <set> is an expression involving "*" and/or "?," then the variable is set to each matching pattern from the input data. In this case, only one such <item> may be in the set, and any <item> besides the first is ignored.

NOTE: The words IN, FOR, and DO must be entered in uppercase.

Examples are

```
FOR %%f IN (*.ASM) DO MASM %%f;
FOR %%f IN (FOO BAR BLECH) DO REM %%f
```

The "%%" is needed so that after batch parameter (%0-%9) processing is done, there is one "%" left. If only "%f" were there, the batch parameter processor would see the "%", look at "f", decide that "%f" was an error (bad parameter reference) and throw out the "%f", so that the FOR command would never see it. If the FOR is not in a batch file, then only one "%" should be used.

FORMAT Command

Syntax:

```
FORMAT [d:][/S][/V]
```

Type:

External

Function:

Formats the diskette in the specified diskette drive so that MS-DOS files can be stored on it.

Comments:

The FORMAT command initializes the directory and file allocation tables. If no drive is specified, the diskette in the default drive is formatted.

If the /S switch is specified, FORMAT copies operating system files from the diskette in the default drive to the newly formatted diskette. The files are copied in the following order:

IO.SYS

MSDOS.SYS

COMMAND.COM

The /V switch option allows you to assign an 11-character volume label, or name, to a diskette. When /V is specified, FORMAT prompts

```
Volume label (11 characters, ENTER for none)?
```

You can enter up to 127 characters. However, MS-DOS only uses the first eleven characters entered for the volume label.

FORMAT operates only on diskettes. To format a hard disk, you must use the HDFORMAT command.

GOTO Command

Syntax:

GOTO <label>

Type:

Internal

Function:

Executes a command after a label in batch file processing.

Comments:

GOTO causes commands to be taken from the batch file beginning with the line after the <label> definition. If no label has been defined, processing of the current batch file will terminate.

For example, for the label "foo" in a batch file

```
:foo
REM looping...
GOTO foo
```

will produce an infinite sequence of the messages, "REM looping...".

When ":" begins a text line in a batch file, that line is where the label to be referenced by a GOTO command is supposed to be entered.

The characters following GOTO usually define a label, but this command may also be used to put in comment lines.

HDFORMAT Command

Syntax:

HDFORMAT

Type:

External

Function:

Prepares a hard disk for data storage and operates on the "volumes" of storage space that it creates on this device.

Comments:

The HDFORMAT command prepares a hard disk for the storage of your MS-DOS files by formatting it, then by partitioning it into volumes, or units of storage space.

HDFORMAT also maintains the hard disk by allowing you to assign data to alternate tracks when tracks become defective. You can change the names of volumes, remove and add volumes, and perform other operations on volumes using sub-commands.

HDFORMAT begins by attempting to access configuration (number of data blocks and tracks on the disk, starting location of data, number of volumes allocated, and so forth) on the hard disk. If no valid configuration information is available, a default hard disk configuration is assumed and messages similar to the following appear.

```
The drive configuration cannot be read,
a new configuration will be created.
Type space to continue.
```

Pressing SPACE causes the program to continue. The hard disk will require formatting before it is suitable for data storage. See instructions for the F(ormat sub-command. If the disk is formatted, then the main command prompt appears:

```
Command: B(ad track, V(olumes, F(ormat, E(xit, U(pdate [A]
```

along with a blank display format.

```
Name    System    Status    Start    End    Blocks    Unused

A)
B)
C)
D)
E)
F)
G)
H)
I)

There are 13 blocks per track. There are 8 tracks reserved for bootstraps, 1400
tracks for data, and 32 tracks for alternates.

Bad Alternate        Bad Alternate        Bad Alternate        Bad Alternate

Copyright (c) Ticom Systems, 1983 All rights reserved
```

The display consists of two sections: the volumes information and the bad tracks information.

The volumes information is maintained in the upper half of the screen and shows how the hard disk is partitioned into logical volumes. The bad tracks information is maintained in the lower half of the screen. It displays a list of defective tracks and their replacement tracks.

One volume may be described on each line of the volumes information section. Each line is identified by a single letter at the left margin. Several volume attributes are shown, including a descriptive name, a volume type, a current status, a disk location, and a volume size. The descriptive name is a user-defined string of up to 19 characters whose sole function is to describe the volume to someone viewing the volumes information. The volume type indicates which system (either the p-System or MS-DOS) can access the volume.

NOTE

HDFORMAT can be used to prepare a hard disk drive for file storage in a p-System operating system. Volumes created for use with p-System cannot be accessed, however, by MS-DOS, or vice versa.

The volume status indicates whether the volume is mounted or dismounted. A volume is accessible to MS-DOS only if it is mounted. The disk space occupied by a volume is displayed as a starting track and an ending track; the volume occupies all tracks between, and including, the starting and ending tracks. A track is a group of 13/512-byte blocks. A volume, therefore, contains an integral multiple of 13 blocks. The number of blocks contained in a volume is calculated and displayed, in addition to the number of unallocated blocks between the end of the volume and the beginning of the next.

A summary of the hard disk allocation information: the size of a track, the bootstrap area, the volume data area, and the alternate track area are displayed below the volumes information.

The bad track information in the lower half of the screen shows defective track numbers and the alternate numbers you have assigned them using the B(ad tracks sub-command.

Note that sub-command prompts contain a list of command options separated by commas. Each sub-command option consists of a command name whose first letter is capitalized and is separated from the rest of the command name by an open parenthesis ("("). A sub-command is invoked by typing the first letter or its name. A command to exit the option and return to the previous prompt is always provided (usually the E(xit command).

The following sections describe the HDFORMAT sub-commands.

B(ad Tracks

The B(ad tracks sub-command is used to specify defective tracks for HDFORMAT to assign alternate tracks. This sub-command is invoked by typing B at the main command prompt. The cursor appears at the first available field in the bad tracks information. Pressing return cancels the sub-command.

To assign alternate tracks, enter the number of a defective track. HDFORMAT searches for a reliable alternate tracks, transfers all recoverable data from the defective track to the alternate tracks, then updates the bad track information. The bad track display is maintained in order of ascending track number. Henceforth, all accesses to the defective track are automatically diverted to the alternate track.

NOTE

Defective tracks are automatically identified and reassigned by F(ormat), but from time to time, defective tracks are found that were missed or that have become defective since the hard disk was originally formatted. These tracks may be identified when a read from the hard disk results in an error. The number of the track containing a bad block may be calculated using the following formula:

bad block number DIV 13 + volume's starting track

The volumes information contains the starting track for each volume.

V(olumes

The V(olumes sub-command allows access to further sub-commands. These sub-commands add, change, remove, mount, dismount, and initialize volumes. In addition, entire volume configurations may be loaded from or dumped to a data file. Typing V at the main command prompt causes the volumes prompt to be displayed.

```
Volumes: A(dd, C(hange, R(emove, M(ount, D(ismount,
I(nit, L(oad, W(rite, E(xit
```

Typing E for E(xit returns to the main command prompt. A(dd, C(hange, R(emove, M(ount, and D(ismount affect individual lines in the volumes information. The L(oad and W(rite commands affect the entire volumes information.

A(dd

The A(dd sub-command allows the addition of a new volume. Typing A causes HDFORMAT to prompt for the name of the new volume on the first unused line of the volumes information. Up to 19 characters describing the new volume may be entered.

Next, a prompt for the volume type is displayed. By default, volumes are p-System volumes. Typing either P or p changes the volume to a p-System volume. Typing either M or m changes it back to an MS-DOS volume.

Next, a prompt for the volume status is displayed. The default status is "Mounted." Typing either D or d changes the status to "Dismounted." Typing either M or m changes it back to "Mounted." Note that if the number of existing volumes currently mounted is equal to the maximum number of mounted volumes for the new volume's system (3 for the p-System, 4 for MS-DOS), no status prompt is generated and the new volume is automatically dismounted.

Next, a prompt for the starting track number is displayed. The starting track number is the number of the lowest track to be occupied by the new volume. It should be an unallocated track between 8 and 1407. Unallocated tracks can be determined by examining the Unused column of the volumes display. Non-zero values in this column indicate empty tracks after the track number listed in the End column on the same line. The value in the Unused column indicates the number of blocks available in the unallocated space. Note that if the first entry in the volumes information does not start on track 8, a volume may be created starting on track 8.

Next, a prompt for the volume size is displayed. The volume size is specified as a number of 512 byte blocks. The default volume size is the number of unallocated blocks between the starting track and the beginning of the next volume. Any volume size entered that does not correspond to an integral number of tracks is automatically adjusted upward to specify an integral number of tracks.

Finally, a prompt is displayed asking whether or not to initialize the new volume.

```
Initialize this volume (Y/N)?
```

Pressing either N, RETURN, or SPACE skips volume initialization. Typing either Y or y causes volume initialization to proceed. An empty root directory and File Allocation Table are written onto the new volume. Then, a prompt is displayed requesting a root directory name.

```
What is the new volume name for entry C (esc to escape) ?
```

A volume name may be up to 11 characters long. If you do not specify a volume name, the first 11 characters of the volume description is the default volume name. Any embedded :'s, $'s, and ='s are removed. Pressing ESC or entering an empty directory name aborts volume initialization; otherwise, a new root directory is written onto the new volume.

C(hange

The C(hange sub-command allows the modification of certain attributes of an existing volume. Typing C causes HDFORMAT to prompt for a letter identifying the line of the volumes information to modify. Typing a valid line identifier causes HDFORMAT to prompt for various fields on the specified line. The entries acceptable under C(hange are the same as those for A(dd. Any current values are treated as the default responses to the prompts and are automatically displayed.

Note that if the volume is an MS-DOS volume, it is not possible to change the volume size. If the volume is the p-System volume, the volume size can be changed.

R(emove

The R(emove sub-command allows the removal of an existing volume, thus releasing its tracks for future use. Typing R causes HDFORMAT to prompt for a letter identifying the line to erase from the volumes information. Typing a valid line identifier causes HDFORMAT to verify the removal.

```
Removing entry C; are you sure (Y/N)?
```

Pressing either N, n, RETURN, or SPACE cancels the operation. Pressing either Y or y causes the specified display line to be erased and any subsequent lines to be moved up.

NOTE

> A volume that is mistakenly R(emoved may be recovered as long as no other volume has been created over or extended onto the tracks occupied by the deleted volume. A deleted volume may be recovered by A(dding a new volume that starts where the old one started and occupies the same number of blocks. This new volume should not be initialized, because the initialization process destroys an old directory by overwriting it with an empty one.

M(ount and D(ismount

The M(ount and D(ismount sub-commands allow modifications to the status attribute of a volume entry. Typing either M for M(ount or D for D(ismount causes HDFORMAT to prompt for a letter identifying a line in the volumes information. Mounting a volume makes the volume accessible to MS-DOS. Dismounting a volume makes the volume inaccessible. Note that a volume may be mounted or dismounted using C(hange but the M(ount and D(ismount sub-commands are faster and more convenient.

I(nit

The I(nit sub-command allows an empty directory to be written on a volume without having to R(emove and re-A(dd the volume. Typing I causes HDFORMAT to prompt for a letter identifying a volume to initialize in the volumes information. Typing a valid line identifier causes HDFORMAT to verify the initialization.

```
Initializing entry A: ; are you sure (Y/N)?
```

Pressing either N, n, RETURN or the SPACE cancels the initialization. Pressing either Y or y proceeds with the operation. If the specified volume is an MS-DOS volume, a new directory is written onto it. If the specified volume is a p-System then, a prompt is displayed requesting a directory name.

```
What is the new volume name for entry C (esc to escape)?
```

A volume name may be up to 11 characters long. Any embedded :'s, $'s, and ='s are removed. Pressing ESC or entering an empty volume name cancels volume initialization; otherwise, a new root directory is written to the volume.

W(rite and L(oad

The W(rite and L(oad sub-commands allow the storage and retrieval of volume configuration information in data files. Typing either W or w for W(rite, or L or l for L(oad causes HDFORMAT to prompt for the name of a data file. The default data file name is NEC.Vol. The W(rite sub-command writes the current volumes display to the specified data file. The L(oad command reads a volume's configuration information from the specified data file and displays it.

F(ormat

The F(ormat sub-command is used to prepare a new hard disk drive for data storage or to rejuvenate a damaged hard disk drive. Since the formatting process destroys any data that may already exist on a disk, the following prompt is displayed.

```
Formatting may destroy data; are you sure (Y/N)?
```

Pressing either N, n, ESC, RETURN or SPACE cancels the F(ormat command. When you press either the Y or y, F(ormat asks for the starting track number.

```
What is the starting track (esc to escape)?
```

The default starting track is track 0; pressing RETURN causes track 0 to become the starting track. Next, a prompt for the ending track is displayed.

```
What is the ending track (esc to escape)?
```

The default ending track is the number of the last track on the disk. Pressing RETURN causes this track to become the ending track. To designate other than the last track on the disk as the ending track, type its number. Note that the ending track number must not be lower than the starting track number.

Formatting begins after the ending track is specified. First, each track in the specified range is formatted. The current track number is displayed in the upper left corner of the screen. Second, each track is the specified range is checked for reliability. Third, any tracks found to be incapable of reliable data storage are marked defective and alternate tracks are automatically allocated. Finally, the bad tracks display is updated to reflect the addition of any new defective tracks.

NOTE

Formatting tracks marked defective does not deallocate any alternate track that may have been assigned to it. Once a track is designated defective, it is not redeemable.

E(xit

The E(xit sub-command terminates HDFORMAT without updating the configuration information area on the hard disk. E(xit may be used to cancel any changes that have been made to either the volumes or bad blocks information. Note that E(xit does not cancel the format and volume initialization operations.

U(pdate

The U(pdate sub-command terminates HDFORMAT after updating the configuration information area on the hard disk.

IF Command

Syntax:

IF <condition> <command>

Type:

Internal

Function:

Executes a command in batch file processing based on a truth test.

Comments:

The parameter <condition> is one of the following:

ERRORLEVEL <number>

True if and only if the previous program executed by COMMAND had an exit code of <number> or higher.

<string1> == <string2>

True if and only if <string1> and <string2> are identical after parameter substitution. Strings may not have embedded separators.

EXIST <filename>

True if and only if <filename> exists.

NOT <condition>

True if and only if <condition> is false.

The IF statement allows the conditional execution of commands. When the <condition> is true, then the <command> is executed. Otherwise, the <command> is ignored.

Note that the words ERRORLEVEL, EXIST, and NOT must be entered in uppercase.

Two examples are

```
IF NOT EXIST TMP FOO ECHO Can't find file
IF NOT ERRORLEVEL 3 LINK $1,,;
```

KEY Command

Syntax:

KEY

Type:

External

Function:

Creates and assigns a series of data strings to the APC dual-mode function keys PF1 through PF16.

Comments:

The KEY command allows you to create and assign a series of data strings to the dual-mode function keys PF1 through PF16. The function key definitions are stored in either the default FUNCTION.KEY file or in files with the .KEY file name extension. KEY allows you to create, display, change, and delete the function key definitions. The program also allows you to change the currently loaded function key assignments.

A total of 32 functions can be defined in each .KEY file (or in FUNCTION.KEY) considering the shifted function of each key. The shifted function of PF1 through PF16 is accessed by simultaneously pressing the function key and FNC.

KEY has two functions: 1) loading a set of existing function key assignments from the storage device, and 2) updating or creating a function key file. After you enter the command line as shown above, the following prompt appears:

```
SOFT KEY DEFINITION PROGRAM V1.1
  UPDATE OR LOAD (U,L)?
```

If you enter L, you must then identify the input filename in one of the following formats:

- file name
- d:filename

If you press only RETURN, KEY automatically loads the current function key assignments stored in the default FUNCTION.KEY file.

If you type U to update a function key file, the filename you enter or the input file determines whether you will be maintaining an existing file or creating a new one. If the filename entered does not exist, KEY creates a file by the name entered on the d: drive, or default drive.

If you press only RETURN, the default input file is FUNCTION.KEY.

An output file specification must then be entered. The output file stores the new or changed function key set. The entry format is the same as for the input file; if you press only RETURN, the output file is the same as the input file.

KEY has the sub-commands listed in Table 4-7.

Table 4-7 KEY Sub-Commands

SUB-COMMAND	ACTION
F ##, [XXXXXXXXXXXXXXX]	Assigns a function to a particular key. The ## identifies the number of the key (1-16) you are assigning the function to. The optional F before the number indicates there is also a function assigned to the key in its shifted mode. The X's represent the actual function assigned, expressed in no more than 15 characters.
D	Displays all function key assignments currently in memory.
A	Initializes all function key assignments currently in memory.
E	Ends function key assignment. You then have the option of updating the default key assignments in FUNCTION .KEY using the current assignments.

In addition to data strings, you can also include standard ASCII and MS-DOS control codes within function key assignments. To invoke a carriage return as a part of the data string, use the character.

The following represent valid examples of function key assignments that you can specify after you enter input and output file specifications.

```
1,COPY B:*.* *
```

The character performs a carriage return after the command COPY B:*.* is entered by pressing PF1.

```
F16,KEY
```

The character performs a carriage return after the KEY command is entered by pressing FNC and PF16 simultaneously.

LOCATE Command

Syntax:

LOCATE <filespec> [d:][<filename>[<.ext>]]

Type:

External

Function:

Converts .EXE (executable) program files to binary format. This results in a saving of device space and faster program loading.

Comments:

This command is useful only if you want to convert .EXE files to binary format. The file named by filespec is the input file. If no extension is specified, it defaults to .EXE. The input file is converted to .COM file format (memory image of the program) and placed in the output file. If you do not specify a drive, the drive of the input file will be used. If you do not specify an output filename, the input filename will be used. If you do not specify a filename extension in the output filename, the new file will be given an extension of .BIN.

The input file must be in valid .EXE format produced by the linker. The resident part, or actual code, and data part of the file must be less than 64K. There must be no STACK segment.

Two kinds of conversions are possible, depending on whether the initial CS:IP (Code Segment:Instruction Pointer) is specified in the .EXE file.

1. If CS:IP is not specified in the .EXE file, a pure binary conversion is assumed. If segment fixups are necessary (that is, the program contains instructions requiring segment relocation), you will be prompted for the fixup value. This value is the absolute segment at which the program is to be loaded. The resulting program will be usable only when loaded at the absolute memory address specified by a user application. The command processor will not be capable of properly loading the program.

2. If CS:IP is specified as 0000:100H, it is assumed that the file is to be run as a .COM file with the location pointer set at 100H by the assembler statement ORG; the first 100H bytes of the file are deleted. No segment fixups are allowed, as .COM files must be segment relocatable; that is, they must assume the entry conditions explained in the *MS-DOS System Reference Guide*. Once the conversion is complete, you may rename the resulting file with a .COM extension. Then the command processor will be able to load and execute the program in the same way as the .COM programs supplied on your MS-DOS diskette.

If CS:IP does not meet either of these criteria, or if it meets the .COM file criterion but has segment fixups, the following message will be displayed:

```
File cannot be converted
```

This message is also displayed if the file is not a valid executable file.

If LOCATE finds an error, one or more of the error messages in Table 4-8 will be displayed.

Table 4-8 LOCATE Error Messages

ERROR	CORRECTIVE PROCEDURE
File not found	The file is not on the drive specified.
Insufficient memory	There is not enough memory to run LOCATE.
File creation error	LOCATE cannot create the output file. Run CHKDSK to determine if the directory is full, or if some other condition caused the error.
Insufficient disk space	There is not enough diskette or hard disk space to create a new file.
Fixups needed — base segment (hex)	The source (.EXE) file contained information indicating that a load segment is required for the file. Specify the absolute segment address at which the finished module is to be located.
File cannot be converted	The input file is not in the correct format.
WARNING — Read error on EXE file. Amount read less than size in header	This is a warning message only.

MKDIR Command

Syntax:

```
MKDIR <pathname>
```

Synonym:

```
MD
```

Type:

Internal

Function:

Makes a new directory.

Comments:

The MKDIR command is used to create a hierarchical directory structure. When you are in your root directory, you can create subdirectories by using the MKDIR command. The command

```
MKDIR \USER
```

will create a subdirectory \USER in your root directory. To create a directory named JOE under \USER, enter

```
MKDIR \USER\JOE
```

MORE Command

Syntax:

```
MORE
```

Type:

External

Function:

Sends output to the console one screen at a time.

Comments:

MORE is a filter that reads from standard input, such as a command from your keyboard, and displays one screen of information at a time. The MORE command then pauses and displays the --MORE-- message at the bottom of your screen.

Pressing RETURN will display another screen of information. This process continues until all the input data has been read.

The MORE command is useful for viewing a long file one screen at a time. If you type

```
TYPE MYFILES.COM | MORE
```

MS-DOS will display the file MYFILES.COM on the default drive one screen at a time.

You can cancel the display operation by pressing CTRL-C. Pressing CTRL-C redisplays the system prompt.

PATH Command

Syntax:

PATH [<pathname>[;<pathname>]...]

Type:

Internal

Function:

Sets a command path.

Comments:

This command allows you to tell MS-DOS which directories should be searched for external commands after MS-DOS searches your working directory.

For example, to tell MS-DOS to search your \BIN\USER\JOE directory for external commands, enter

```
PATH \BIN\USER\JOE
```

for searching root dir; path \ [illegible]

You can tell MS-DOS to search more than one path by specifying several pathnames separated by semicolons. For example,

```
PATH \BIN\USER\JOE;\BIN\USER\SUE;\BIN\DEV
```

tells MS-DOS to search the directories specified by the above pathnames to find external commands. MS-DOS searches the pathnames in the order specified in the PATH command.

The command PATH with no options will print the current path. If you specify PATH ;, MS-DOS will set the NUL path, meaning that only the working directory will be searched for external commands.

PAUSE Command

Syntax:

PAUSE [comment]

Type:

Internal

Function:

Suspends execution of a batch file.

Comments:

During the execution of a batch file, you may need to change diskettes or perform some other action. PAUSE suspends execution until you press any key, except CTRL-C.

When the command processor encounters PAUSE, it prints

```
Strike a key when ready...
```

If you press CTRL-C, another prompt will be displayed:

```
Abort batch job (Y/N)?
```

If you type "Y" in response to this prompt, execution of the remainder of the batch command file will be cancelled and control will be returned to the operating system command level. Therefore, PAUSE can be used to break a batch file into pieces, allowing you to end the batch command file at an intermediate point.

The comment is optional and may be entered on the same line as PAUSE. You can use the comment to prompt the user of the batch file with some meaningful message when the batch file pauses. The comment prompt will be displayed before the "Strike a key" message.

POW Command

Syntax:

POW

Type:

External

Function:

Turns off the main power supply of the APC.

Comments:

The POW command can be incorporated into a batch processing job stream for the purpose of shutting down the system without operator intervention at the end of processing.

To restore power to the APC, turn the main power switch to the OFF position, then to the ON position.

PRINT Command

Syntax:

```
PRINT [[filespec][/T][/C][/S]]...
```

Type:

External

Function:

Prints the contents of a file on a line printer while you are processing other MS-DOS commands. This process is usually called "background printing."

Comments:

You will use the PRINT command only if you have a line printer attached to your computer. The following switches are provided with this command:

/T	TERMINATE: This switch deletes all files in the print queue (those waiting to be printed). A message to this effect will be printed.
/C	CANCEL: This switch turns on cancel mode. The preceding filespec and all following filespecs will be suspended in the print queue until you type a /P switch.
/P	PRINT: This switch turns on print mode. The preceding filespec and all following filespecs will be added to the print queue until you issue a /C switch.

PRINT with no options displays the contents of the print queue on your screen without affecting the queue.

Examples:

```
PRINT /T
```

empties the print queue.

```
PRINT /T *.ASM
```

empties the print queue and queues all .ASM files on the default drive.

```
PRINT A:TEMP1.TST/C A:TEMP2.TST A:TEMP3.TST
```

removes the three files indicated from the print queue.

```
PRINT TEMP1.TST /C TEMP2.TST /P TEMP3.TST
```

removes TEMP1.TST from the queue, and adds TEMP2.TST and TEMP3.TST to the queue.

If an error is detected, PRINT will display one of the error messages listed in Table 4-9.

Table 4-9 PRINT Error Messages

ERROR	CORRECTIVE PROCEDURE
Name of list device [PRN:]	This prompt appears when PRINT is run for the first time. Any current device may be specified and that device then becomes the PRINT output device. As indicated in the [ ], simply pressing RETURN results in the device PRN being used.
List output is not assigned to a device	This message will be displayed if the "Name of list device" specified to the above prompt is invalid. Subsequent attempts will return the same message until a valid device is specified.
PRINT queue is full	There is room for 10 files in the queue. If you attempt to put more than 10 files in the queue, this message will appear on the console.
PRINT queue is empty	There are no files in the print queue.
No files match d:XXXXXXXX.XXX	A filespec was given for files to add to the queue, but no files match a specification. Note if there are no files in the queue to match the canceled filespec, no error message will appear.

Table 4-9 PRINT Error Messages (cont'd)

ERROR	CORRECTIVE PROCEDURE
Drive not ready	If this message occurs when PRINT attempts to access a storage device, PRINT will keep trying until the drive is ready. Any other error causes the current file to be canceled. An error message would be output on your printer in such a case.
All files canceled	If the /T (TERMINATE) switch is issued, the message "All files canceled by operator" will be printed on your printer. If the current file being printed is canceled by a /C, the message "File canceled by operator" will be printed.

PROMPT Command

Syntax:

PROMPT [<prompt-text>]

Type:

Internal

Function:

Changes the MS-DOS command prompt.

Comments:

The PROMPT command allows you to change the MS-DOS system prompt. If no text is typed, the prompt will be set to the default prompt, which is the default drive designation. You can set the prompt to a special prompt, such as the current time, by using the characters indicated below.

The characters in Table 4-10 can be used in the prompt command to specify your own prompts. They must all be preceded by a dollar sign ($) in the prompt command.

Table 4-10 User Prompt Characters

CHARACTER	PROMPT
t	The current time
d	The current date
p	The current directory of the default drive
v	The version number
n	The default drive
g	The > character
l	The < character
b	The | character
-	A carriage return and line feed sequence
s	A space
h	A backspace
e	ASCII code X'1B' (escape sequence)

Examples:

```
PROMPT   $n
```

Sets the normal MS-DOS prompt (:).

```
PROMPT   Time = $t$_Date = $d
```

Sets a two-line prompt which prints:
Time = (current time)
Date = (current date).

You can use escape sequences in your prompts. For example,

```
PROMPT   $e[7m$n:$e[m
```

Sets the prompt in inverse video mode and returns to video mode for other text.

RDCPM Command

Syntax:

```
RDCPM   filespec [filespec]
        DIR filespec
        DIR d:
```

Type:

External

Function:

Reads a file from a single-density diskette that has been formatted on a CP/M system and copies that file to a different diskette assumed to be formatted with MS-DOS. The directory of the CP/M formatted diskette is displayed.

Comments:

The RDCPM command is similar to the COPY command except that the source file specified with the first filespec is assumed to be on a CP/M diskette. The alternate RDCPM syntax with DIR allows the directory of the CP/M diskette to be displayed.

```
RDCPM DIR filespec
RDCPM DIR <d:>
```

Wildcard characters ("*" and "?") are acceptable in filenames.

RECOVER Command

Syntax:

```
RECOVER <filename | d:>
```

Type:

External

Function:

Recovers a file on a diskette or an entire diskette containing bad sectors.

Comments:

If a sector on a diskette is bad, you can recover either the file containing that sector (without the bad sector) or the entire diskette (if the bad sector was in the directory).

To recover a particular file, enter

```
RECOVER <filename>
```

This will cause MS-DOS to read the file sector by sector and to skip the bad sector(s). When MS-DOS finds the bad sector(s), it is marked and MS-DOS will no longer allocate your data to that sector.

To recover a diskette, enter

```
RECOVER <d:>
```

where d: is the letter of the drive containing the diskette to be recovered.

If there is not enough room in the root directory, RECOVER will print a message and store information about the extra files in the File Allocation Table. You can run RECOVER again to regain these files when there is more room in the root directory.

REM Command

Syntax:

REM [comment]

Type:

Internal

Function:

Displays remarks which are on the same line as the REM command in a batch file during execution of that batch file.

Comments:

The only separators allowed in the comment are the space, tab, and comma.

The following is an example of a REM command in a batch file.

```
1: REM This file checks new disks
2: REM It is named NEWDISK.BAT
3: PAUSE Insert new disk in drive B:
4: FORMAT B:/S
5: DIR B:
6: CHKDSK B:
```

The numbers at the left are not part of the command. They indicate the text line numbers, which are displayed by EDLIN for entering the commands (see Chapter 5).

REN Command

Syntax:

```
REN <filespec> <filename>
```

Synonym:

RENAME

Type:

Internal

Function:

Changes the name of the first option (filespec) to the second option (filename).

Comments:

The first option (filespec) must be given a drive designation if the file resides on a drive other than the default drive. Any drive designation for the second option (filename) is ignored. The file will remain on the device where it currently is stored.

The wild card characters may be used in either option. All files matching the first filespec are renamed. If wild card characters appear in the second filename, corresponding character positions will not be changed.

For example, the following command changes the names of all files with the LST extension to similar names with the PRN extension.

```
REN *.LST *.PRN
```

In the next example, REN renames the file ABODE on drive B to ADOBE.

```
REN B:ABODE ?D?B?
```

The file remains on drive B.

An attempt to rename a filespec to a name already present in the current directory will result in the error message "File not found."

RMDIR Command

Syntax:

```
RMDIR <pathname>
```

Synonym:

```
RD
```

Type:

Internal

Function:

Removes a directory from a hierarchical directory structure.

Comments:

This command removes a directory that is empty except for the . and .. shorthand symbols.

For example, to remove the \BIN\USER\JOE directory, first issue a DIR command for that path to ensure that the directory does not contain any important files that you do not want deleted. Then enter

```
RMDIR \BIN\USER\JOE
```

The directory has been deleted from the directory structure.

SET Command

Syntax:

SET [<string = string>]

Type:

Internal

Function:

Sets one string value equivalent to another string for use in later programs.

Comments:

This command is meaningful only if you want to set values that will be used by programs you have written. An application program can check all values that have been set with the SET command by issuing SET with no options. For example, SET TTY = VT52 sets your TTY value to VT52 until you change it with another SET command.

The SET command can also be used in batch processing. In this way, you can define your replaceable parameters with names instead of numbers. If your batch file contains the statement "LINK %FILE%," you can set the name that MS-DOS will use for that variable with the SET command. The command SET FILE = DOMORE replaces the %FILE% parameter with the filename DOMORE. Therefore, you do not need to edit each batch file to change the replaceable parameter names. Note that when you use text (instead of numbers) as replaceable parameters, the name must be ended by a percent sign.

SETCOM Command

Syntax:

```
SETCOM
SETCOM speed, length, parity, stop bit
SETCOM,
```

Type:

External

Function:

Sets the working conditions of the RS-232-C serial interface of the APC.

Comments:

Parameters are set according to the requirements of the device with which the APC is communicating. The parameters can be invoked directly in the command line as shown in the second syntax format above, or after SETCOM is loaded using the first syntax format of the command line. The third syntax format automatically sets the default parameters described.

The SETCOM parameter options control four areas affecting communication through the RS-232-C interface. Options must be entered in the order shown above with commas separating each parameter. If a parameter is omitted, the comma must be entered with a blank for the option. In this case, the default value is automatically assigned.

• speed	Sets the baud rate (bits/second) of the data transfer. Any of the following values is acceptable: 200, 300, 600, 1200, 2400, 9600, or 19200. The default speed is 300 bits/second.
• length	Sets the length (in bits) of the data word being transferred. Any of the following values is acceptable: 5, 6, 7, or 8. The default word length is 7 bits.
• parity	Sets the type of parity required for the data being transferred. Even parity is set by entering E; odd parity is set by entering O; no parity check is set by entering N. The default parameter for parity is E.
• stop bit	Sets the number of stop bits required during transfers. Two values, 1 or 2, are acceptable with the default parameter being 1.

For example,

```
A:SETCOM 2400,8,N,2
```

sets a baud rate of 2400 bits/second for transfer of data words 8 bits long. No parity check is required, but 2 stop bits are required.

The command line

```
A:SETCOM 2400,8,,2
```

sets the same conditions for serial transfer as the previous example, except the parity is set to E by using the default value (represented by the blank).

SHIFT Command

Syntax:

SHIFT

Type:

Internal

Function:

Allows access to more than 10 replaceable parameters in batch file processing.

Comments:

Usually, command files are limited to handling 10 parameters, %0 through %9. To allow access to more than ten parameters, use SHIFT to change the command line parameters. For example,

```
if  %0 = "foo"
    %1 = "bar"
    %2 = "name"
    %3...%9 are empty
```

then a SHIFT will result in the following:

```
    %0 = "bar"
    %1 = "name"
    %2...%9 are empty
```

If there are more than 10 parameters given on a command line, those that appear after the 10th (%9) will be shifted one at a time into %9 by successive shifts.

SORT Command

Syntax:

```
SORT [/R] [/+n]
```

Type:

External

Function:

SORT reads standard input (from your keyboard to the screen), sorts the data, then writes it back to your screen or to files.

Comments:

SORT can be used, for example, to alphabetize input data by a certain column. There are two switches that allow you to select options:

/R	reverses the sort; that is, sorts from Z to A.
/+n	sorts starting with column n where n is some number. If you do not specify this switch, SORT will begin sorting from column 1.

For example, the following command will pipe the output of the directory command to the SORT filter. The SORT filter will sort the directory listing starting with column 14 (this is the column in the directory listing that contains the file size), then send the output to the console. Thus, the result of this command is a directory sorted by file size.

```
DIR | SORT /+14
```

The command

```
DIR | SORT /+14 | MORE
```

will do the same thing as the command in the previous example, except that the MORE filter will give you a chance to read the sorted directory one screen at a time.

If you want to process the data in a file, you must use the TYPE command to pipe the data to SORT:

```
TYPE FILE.DAT| SORT
```

Alternatively, SORT may take data from a file using the redirection of input feature.

```
SORT <FILE.DAT
```

sort < file.dat > file.sor
must have space

SORT /R <UNSORT.TXT >SORT.TXT Reverse order

DIR | SORT /+14 Pipes directory to sort, dir is sorted on col. 14 filesize, Out put is sent to std input device

SYS Command

Syntax:

SYS <d>:

Type:

External

Function:

Transfers the MS-DOS program files from the diskette in the default drive to the diskette in the drive specified by d:.

Comments:

SYS is normally used to update the MS-DOS system files as supplied by NECIS. An entry for d: is required. The diskette in drive d: should have been formatted using the FORMAT command without the /S switch.

If IO.SYS and MSDOS.SYS are already on the destination diskette, they must take up the same amount of space on the diskette as the new system will need. This means that you cannot transfer system files from an MS-DOS 2.0 diskette to an MS-DOS 1.1 diskette. You must reformat the MS-DOS 1.1 diskette with the FORMAT command before the SYS command will work.

The destination diskette must be completely blank or already have the system files IO.SYS and MSDOS.SYS.

The transferred files are copied in the following order:

IO.SYS
MSDOS.SYS

COMMAND.COM (the command processor) is not transferred. You must use the COPY command to transfer COMMAND.COM.

If SYS detects an error, one of the messages in Table 4-11 will be displayed.

Table 4-11 SYS Error Messages

ERROR	CORRECTIVE PROCEDURE
No room for system on destination diskette	There is not enough room on the destination diskette for the IO.SYS and MSDOS.SYS files.
Incompatible system size	The system files IO.SYS and MSDOS.SYS do not take up the same amount of space on the destination diskette as the new system will need.

Note that you cannot use SYS to copy the MS-DOS programs to hard disk.

TIME Command

Syntax:

```
TIME [<hh> [:<mm> ]]
```

Type:

Internal

Function:

Displays and sets the time.

Comments:

If the TIME command is entered without any arguments, the following message is displayed:

```
Current time is <hh>:<mm>:<ss>.<cc>
Enter new time:_
```

Press RETURN if you do not want to change the time shown. A new time may be given as an option to the TIME command, as in

```
TIME 8:20
```

The new time must be entered using numerals only; letters are not allowed. The allowed options are

<hh> = 00-24
<mm> = 00-59

The hour and minute entries must be separated by colons. You do not have to type the <ss> (seconds) or <cc> (hundredths of seconds) options.

MS-DOS uses the time entered as the new time if the options and separators are valid. If the options or separators are not valid, MS-DOS displays the message,

```
Invalid time
Enter new time: _
```

MS-DOS then waits for you to type a valid time.

TIMEROFF Command

Syntax:

TIMEROFF

Type:

External

Function:

Turns off the dynamic update of the time and date in the APC status line at the top of the screen.

Comments:

The TIMEROFF command turns off the dynamic update of the time and date in the APC status line at the top of the screen. TIMEROFF can be run prior to programs in which the interrupt caused by the update of the status line display results in significant delays in processing. If you are using the RS-232-C interface of the APC for serial communications, run TIMEROFF prior to those communications where timing below approximately 200 milliseconds occurs. Note, however, that because most communications software packages offered for use on the APC already perform this function, you may not need to use TIMEROFF.

Although the status line display is not updated following TIMEROFF, the internal clock/calendar of the APC continues to correctly store the time and date. The dynamic display of the correct time and date is restored by the TIMERON command.

TIMERON Command

Syntax:

TIMERON

Type:

External

Function:

Restores the dynamic display of the time and date in the APC status line.

Comments:

When TIMERON is invoked at the command line as shown above, the correct time and date appear on the APC status line, regardless of the interval between running TIMEROFF and TIMERON. TIMERON reverses the operation of TIMEROFF.

TYPE Command

Syntax:

TYPE <filespec>

Type:

Internal

Function:

Displays the contents of a file on the screen.

Comments:

Use this command to examine a file on diskette or hard disk without modifying it. The only formatting performed by TYPE is that tabs are expanded to spaces consistent with tab stops every eighth column. Note that a display of binary files causes control characters (such as CTRL-Z) to be sent to your computer, along with bells, form feeds, and escape sequences.

VER Command

Syntax:

VER

Type:

Internal

Function:

Displays the MS-DOS version number.

Comments:

If you want to know what version of MS-DOS you are using, type VER. The version number will be displayed on your screen.

VERIFY Command

Syntax:

VERIFY [ON | OFF]

Type:

Internal

Function:

Turns the verify switch on or off when writing to diskette or hard disk.

Comments:

This command has the same purpose as the /V switch in the COPY command. If you want to verify that all files are written correctly to a device, you can use the VERIFY command to tell MS-DOS to verify that your files are intact (no bad sectors, for example). MS-DOS will perform a VERIFY each time you write data to a diskette or hard disk. You will receive an error message only if MS-DOS was unable to successfully complete the operation.

VERIFY ON remains in effect until you change it in a program (see the SET VERIFY system call as explained in the *MS-DOS System Reference Guide*), or until you issue a VERIFY OFF command to MS-DOS.

If you want to know what the current setting of VERIFY is, type VERIFY with no options.

VOL Command

Syntax:

VOL [d:]

Type:

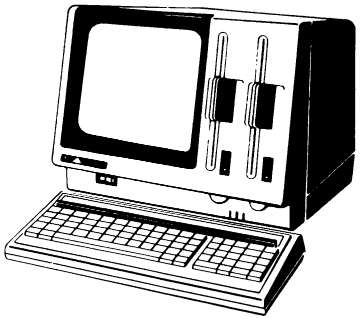
Internal

Function:

Displays the diskette or hard disk volume number, if it exists.

Comments:

This command prints the volume ID of the diskette or hard disk volume in drive d:. If no drive is specified, MS-DOS prints the volume ID of the volume in the default drive.



Chapter 5

The Line Editor (EDLIN)

In this chapter, you will learn how to use EDLIN, the Line Editor program. You can use EDLIN to create, change, and display files, whether they are source program or text files. With EDLIN, you can

- create new source files and save them
- update existing files and save both the updated and original files
- delete, edit, insert, and display lines
- search for, delete, or replace text within one or more lines.

The text in files created or edited by EDLIN is divided into lines, each up to 253 characters long. Line numbers are generated and displayed by EDLIN during the editing process, but are not actually present in the saved file.

When you insert lines in a file, all line numbers following the inserted text advance automatically by the number of lines being inserted. When you delete lines, all line numbers following the deleted text decrease automatically by the number of lines deleted. As a result, lines are always numbered consecutively in your file.

HOW TO START EDLIN

To start EDLIN, enter

```
EDLIN <filespec>
```

If you are creating a new file, the filespec should be the name of the file you want to create. If EDLIN does not find this file on a drive, EDLIN will create a new file with the name you specified. The following message and prompt will be displayed:

```
New file
*_
```

Note that the prompt for EDLIN is an asterisk (*).

You can now type lines of text into your new file. To begin entering text, you must enter an I (Insert) command to insert lines. The I command is discussed later in this chapter.

If you want to edit an existing file, filespec should be the name of the file you want to edit. When EDLIN finds the file you specify on the designated or default drive, the file will be loaded into memory. If the entire file can be loaded, EDLIN will display the following message on your screen:

```
End of input file
*_
```

You can then edit the file using EDLIN editing commands.

If the file is too large to be loaded into memory, EDLIN will load lines until memory is 3/4 full, then display the * prompt. You can then edit the portion of the file that is in memory.

To edit the remainder of the file, you must save some of the edited lines on diskette or hard disk to free memory; then EDLIN can load the unedited lines into memory. Refer to the Write and Append commands in the section on EDLIN commands of this chapter for the save procedure.

When you complete the editing session, you can save the original and the updated (new) files by using the End command (see the section on EDLIN commands). The original file is renamed with an extension of .BAK (for backup), and the new file has the filename and filename extension you specify in the EDLIN command. The original .BAK file will not be erased until the end of the editing session, or until storage space is needed by EDLIN.

Do not try to edit a file with a filename extension of .BAK because EDLIN assumes that any .BAK file is a backup file. If you find it necessary to edit such a file, rename the file with another extension (using the RENAME command described in Chapter 4), then run EDLIN again specifying the new filespec.

EDITING KEYS USED BY EDLIN

The dual-mode function keys and template discussed in Chapter 4 can be used by EDLIN to edit your files.

Descriptions of the functions these keys perform during EDLIN follow in the next pages.

For each key, you are given a name, a statement of its function, and comments that detail the key's operations with examples. In general, text that you have entered on a text line is saved in the template, from where it can be redisplayed in whole or in part by using the dual-mode function keys. The operations of these keys on the text line are actually the same as those for a command line. However, the term "text line" will be used for entered lines of text in the following descriptions of the dual-mode function keys. Note that the number and colon shown at the left of a text line are not part of the text.

Key:

COPY1

Function:

Copies one character from the template to the text line.

Comments:

Pressing COPY1 (PF1) copies one character from the template to the text line. When COPY1 is pressed, one character is inserted in the text line and Insert mode is automatically turned off.

For example, assume that the screen shows

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Pressing COPY1 copies the first character (T) to the second of the two lines displayed:

```
        1:*This is a sample file
COPY1   1:*T
```

Each time COPY1 is pressed, one more character appears:

```
COPY1   1:*Th_
COPY1   1:*Thi_
COPY1   1:*This_
```

Key:

COPYUP

Function:

Copies all the characters up to a specified character.

Comments:

Pressing COPYUP (PF2) copies all characters up to a specified character from the template to the text line. The specified character is the next character typed after COPYUP; it is not copied or displayed on the screen. Pressing COPYUP causes the cursor to move to the single character that is specified in the command. If the template does not contain the specified character, nothing is copied. Pressing COPYUP also automatically turns off Insert mode.

For example, assume that the screen shows

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Pressing COPYUP copies all characters up to the character specified immediately after the COPYUP key.

```
            1:*This is a sample file
COPYUP p    1:*This is a sam_
```

Key:

COPYALL

Function:

Copies the rest of the template to the text line.

Comments:

Pressing COPYALL (PF3) copies all remaining characters from the template to the text line. Regardless of the cursor position at the time COPYALL is pressed, the rest of the line appears, and the cursor is positioned after the last character on the line.

For example, assume that the screen shows

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Pressing COPYALL copies all characters from the template (shown in the upper line displayed) to the line with the cursor (the lower line displayed)

```
             1:*This is a sample file (template)
COPYALL      1:*This is a sample file._ (text line)
```

Also, Insert mode is automatically turned off.

Key:

SKIP1

Function:

Skips over a single character in the template.

Comments:

Pressing SKIP1 (PF4) skips over one character in the template and this character is not copied to the text line. Each time you press SKIP1, one character is not copied from the template. The action of the SKIP1 key is similar to that of the COPY1 key, except that SKIP1 skips a character in the template rather than copying it to the text line.

For example, assume that the screen shows

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Pressing SKIP1 skips over the first character (T).

```
        1:*This is a sample file
SKIP1   1:*_
```

The cursor position does not change and only the template is affected. To see how much of the line has been skipped over, press COPYALL, which moves the cursor beyond the last character of the line.

```
          1:*This is a sample file.
SKIP1     1:*_
COPYALL   1:*his is a sample file._
```

Key:

SKIPUP

Function:

Skips multiple characters in the template up to the specified character.

Comments:

Pressing SKIPUP (PF5) skips over all characters up to a given character in the template and does not copy these characters to the text line. The specified character is not copied and is not shown on the screen. If the template does not contain the specified character, nothing is skipped over. The action of the SKIPUP key is similar to that of the COPYUP key, except that SKIPUP skips over characters in the template rather than copying them to the text line.

For example, assume that the screen shows

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Pressing SKIPUP skips over all the characters in the template up to the character pressed after the SKIPUP key.

```
            1:*This is a sample file
SKIPUP p    1:*_
```

The cursor position does not change. To see how much of the line has been skipped over, press COPYALL to copy the template. This COPYALL moves the cursor beyond the last character of the line.

```
            *This is a sample file:
SKIPUP p    1:*_
COPYALL     1:*ple file._
```

Key:

VOID

Function:

Cancels input and empties the text line.

Comments:

Pressing the VOID (PF6) key empties the text line of characters, but it leaves the template unchanged. VOID also prints a backslash (\), carriage return, and line feed, and turns Insert mode off. The cursor (indicated by the underline) is positioned at the beginning of the line. Pressing COPYALL copies the template to the text line and the text line appears as it was before VOID was pressed.

For example, assume that the screen shows

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Assume that you want to replace the line with "Sample File":

```
1:*This is a sample file.
1:*Sample File_
```

To cancel the line you just entered, and to keep "This is a sample file.", press VOID. Note that a backslash appears on the "Sample File" line to tell you it has been cancelled.

```
        1:*This is a sample file.
VOID    1:*Sample File
        1:_
```

Press RETURN to keep the original line, or to perform any other editing functions. If COPYALL is pressed, the original template is copied to the text line:

```
COPYALL 1: This is a sample file._
```

Key:

INSERT

Function:

Enters Insert mode.

Comments:

Pressing INSERT (PF7) causes EDLIN to enter Insert mode. The current INSERT cursor position in the template is not changed. The cursor does move as each character is inserted. However, when you have finished inserting characters, the cursor will be positioned at the same character as it was before the insertion began. Thus, characters are inserted in front of the character to which the cursor points.

For example, assume that the screen shows

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Assume that you press the COPYUP and f keys.

```
               1:*This is a sample file
COPYALL        1:*This is a sample_
```

Now press INSERT and insert the characters "edit" and a space.

```
               1:*This is a sample file.
COPYUP  f      1:*This is a sample_
INSERT  edit   1:*This is a sample edit_
```

If you now press COPYALL, the rest of the template is copied to the line.

```
               1:*This is a sample edit_
COPYALL        1:*This is a sample edit file._
```

If you pressed RETURN, the remainder of the template would be truncated, or cut off, and the command line would end at the end of the insert.

```
INSERT edit RETURN    1:*This is a sample edit_
```

Key:

REPLACE

Function:

Enters Replace mode.

Comments:

Pressing REPLACE (PF8) causes EDLIN to exit Insert mode and to enter Replace mode. All the characters you type will overstrike and replace characters in the template. When you start to edit a line, Replace mode is in effect. If RETURN is pressed, the remainder of the template will be deleted.

For example, assume that the screen shows

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Assume that you then press COPYUP m, INSERT lary, REPLACE tax, and then COPYALL:

```
              1:*This is a sample file.
COPYUP m      1:*This is a sa_
INSERT lary   1:*This is a salary_
REPLACE tax   1:*This is a salary tax_
COPYALL       1:*This is a salary tax file._
```

Note that you inserted "lary" and replaced "mple" with " tax." If you type characters that extend beyond the length of the template, the remaining characters in the template will be automatically appended (added) when you press COPYALL.

Key:

NEWLINE

Function:

Creates a new template.

Comments:

Pressing NEWLINE (PF9) key copies the current text line to the template. The contents of the old template are deleted. Pressing NEWLINE outputs an @ ("at sign" character), a carriage return, and a line feed. The command line is also emptied and Insert mode is turned off.

NOTE

> NEWLINE performs the same function as VOID, except that the template is changed and an @ ("at sign" character) is printed instead of a \ (back slash).

For example, assume that the screen shows

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Assume that you enter COPYUP m, INSERT lary, REPLACE tax, and then COPYALL:

```
              1:*This is a sample file.
COPYUP m      1:*This is a sa_
INSERT lary   1:*This is a salary_
REPLACE tax   1:*This is a salary tax_
COPYALL       1:*This is a salary tax file._
```

At this point, assume that you want this line to be the new template, so you press NEWLINE:

```
NEWLINE 1:*This is a salary tax file.@
```

The @ indicates that this new line is now the new template. Additional editing can be done using the new template.

EDLIN COMMANDS

EDLIN commands perform editing functions on lines of text.

Information for Entering EDLIN Commands

The following list contains information you should read before you use EDLIN commands.

1. Pathnames are acceptable as options to commands. For example, typing EDLIN \BIN\USER\JOE\TEXT.TXT will allow you to edit the TEXT.TXT file in the subdirectory JOE.

2. You can reference line numbers relative to the current line (the line with the asterisk). Use a minus sign with a number to indicate lines before the current line. Use a plus sign with a number to indicate lines after the current line, for example,

   ```
   -10,+10L
   ```

   This command lists 10 lines before the current line, the current line, and 10 lines after the current line.

3. Multiple commands may be issued on one command line. When you issue a command to edit a single line using a line number (<line>), a semicolon must separate commands on the line. Otherwise, one command may follow another without any special separators. In the case of a Search or Replace command, the string may be ended by a CTRL-Z instead of a RETURN.

   Examples:

   The following command line edits line 15 and then displays lines 10 through 20 on the screen.

   ```
   15;-5,+5L
   ```

   The command line in the next example searches for "This string" and then displays 5 lines before and 5 lines after the line containing the matched string. If the search fails, then the displayed lines are those line numbers relative to the current line.

   ```
   SThis string CTRL-Z -5,+L
   ```

4. You can type EDLIN commands with or without a space between the line number and command. For example, to delete line 6, the command 6D is the same as 6 D.

5. You may insert a control character (such as CTRL-C) into text by using the quote character CTRL-V before it while in Insert mode. CTRL-V tells MS-DOS to recognize the next capital letter typed as a control character. You may also use a control character in any of the string arguments of Search or Replace by using the special quote character. For example,

   S CTRL-V Z

   will find the first occurrence of CTRL-Z in a file

   R CTRL-V Z CTRL-Z foo

   will replace all occurrences of CTRL-Z in a file by foo

   S CTRL-V C CTRL-Z bar

   will replace all occurrences of CTRL-C by bar

   You can insert CTRL-V into the text by typing CTRL-V-V.

6. The CTRL-Z character ordinarily tells EDLIN, "This is the end of the file." If you have CTRL-Z characters elsewhere in your file, you must tell EDLIN that these other control characters do not mean "End of File." Use the /B switch to tell EDLIN to ignore any CTRL-Z characters in the file and to show you the entire file.

The EDLIN commands are summarized in the Table 5-1. They are described in further detail following the description of command options.

Table 5-1 EDLIN Commands

COMMAND	FUNCTION
<line>	Positions you to edit a line
A	Appends lines
C	Copies lines
D	Deletes lines
E	Ends editing, saving edits
I	Inserts lines
L	Lists text
M	Moves lines
P	Pages text
Q	Quits editing, without saving edits
R	Replaces lines
S	Searches text
T	Transfers text
W	Writes lines

Command Options

Several EDLIN commands accept one or more options. The effect of a command option varies, depending on with which command it is used. The following list describes each option.

<line>	<line> indicates a line number that you type to access a line. Line numbers must be separated by a comma or a space from other line numbers, other options, and from the command. <line> may be specified in one of three ways:

Number	Any number less than 65534. If a number larger than the largest existing line number is specified, then <line> means the line after the last line number.
Period	(.) If a period is specified for <line>, then <line> means the current line number. The current line is the last line edited, but is not necessarily the last line displayed. The current line is marked on your screen by an asterisk (*) between the line number and the first character.
Pound	(#) The pound sign indicates the line after the last line number. If you specify # for <line>, this has the same effect as specifying a number larger than the last line number.
RETURN	A carriage return entered without any of the <line> specifiers listed above directs EDLIN to use a default value appropriate to the command.

?	The question mark option directs EDLIN to ask you if the correct string has been found. The question mark is used only with the Replace and Search commands. Before continuing, EDLIN waits for either a ″Y″ or RETURN for a yes response, or for any other key for a no response.
<string>	<string> represents text to be found, to be replaced, or to replace other text. The <string> option is used only with the Search and Replace commands. Each string must be ended by a CTRL-Z or a RETURN (see the Replace command for details). No spaces should be left between strings or between a string and its command letter, unless you want those spaces to be part of the string.

EDLIN COMMAND DESCRIPTIONS

The following pages describe EDLIN commands. You are given the command name, its syntax, a statement of its function, and details of its operations.

Name:

Append

Syntax:

```
[<n>]A
```

Function:

Adds the specified number of lines from diskette or hard disk to the file being edited in memory. The lines are added at the end of lines that are currently in memory.

Comments:

This command is meaningful only if the file being edited is too large to fit into memory. As many lines as possible are read into memory for editing when you start EDLIN.

To edit the remainder of the file that will not fit into memory, lines that have already been edited must be written to the appropriate diskette or hard disk file. Then you can load unedited lines from disk into memory with the Append command. Refer to the Write command in this chapter for information on how to write edited lines to a storage device.

Notes:

1. If you do not specify the number of lines to append, lines will be appended to memory until available memory is 3/4 full. No action will be taken if available memory is already 3/4 full.
2. The message "End of input file" is displayed when the Append command has read the last line of the file into memory.

Name:

Copy

Syntax:

```
[<line>],[<line>],<line>,[<count>]C
```

Function:

Copies a range of lines to a specified line number. The lines can be copied as many times as you want by using the count option.

Comments:

If you do not specify a number in <count>, EDLIN copies the lines once. If the first or the second <line> is omitted, the default is the current line. The file is renumbered automatically after the copy operation.

If the line numbers overlap you will get an "Entry error" message. For example, 3,20,15C would result in an error message.

Examples:

Assume that the following file exists and is ready to edit.

```
1: This is a sample file
2: used to show copying lines.
3: See what happens when you use
4: the Copy command
5: (the C command)
6: to copy text in your file.
```

You can copy this entire block of text by issuing the following command:

```
1,6,7C
```

The result is as follows:

```
 1: This is a sample file
 2: used to show copying lines.
 3: See what happens when you use
 4: the Copy command
 5: (the C command)
 6: to copy text in your file.
 7: This is a sample file
 8: used to show copying lines.
 9: See what happens when you use
10: the Copy command
11: (the C command)
12: to copy text in your file.
```

If you want to place the text within other text, the third <line> option should specify the line before which you want the copied text to appear. For example, assume that you want to copy lines and insert them within the following file:

```
 1: This is a sample file
 2: used to show copying lines.
 3: See what happens when you use
 4: the Copy command
 5: (the C command)
 6: to copy text in your file.
 7: You can also use COPY
 8: to copy lines of text
 9: to the middle of your file.
10: End of sample file.
```

The command 3,6,9C results in the following file:

```
 1: This is a sample file
 2: used to show copying lines.
 3: See what happens when you use
 4: the Copy command
 5: (the C command)
 6: to copy text in your file.
 7: You can also use COPY
 8: to copy lines of text
 9: to the middle of your file.
10: See what happens when you use
11: the Copy command
12: (the C command)
13: to copy text in your file.
14: End of sample file.
```

Name:

Delete

Syntax:

[<line>][,<line>]D

Function:

Deletes a specified range of lines in a file.

Comments:

If the first <line> is omitted, that option will default to the current line (the line with the asterisk next to the line number). If the second <line> is omitted, then just the first <line> will be deleted. When lines have been deleted, the line immediately after the deleted section becomes the current line and has the same line number as the first deleted <line> had before the deletion occurred.

For example, assume that the following file exists and is ready to edit.

```
 1: This is a sample file
 2: used to show dynamic line numbers.
 3: See what happens when you use
 4: Delete and Insert
 .
 .
 .
25: (the D and I commands)
26: to edit the text
27:*in your file.
```

To delete multiple lines, type <line>, line D:

```
5,24D
```

The result is as follows:

```
1: This is a sample file
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
6: to edit text
7:*in your file.
```

To delete a single line, type

```
6D
```

The result is as follows:

```
1: This is a sample file
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
6:*in your file.
```

Next, delete a range of lines from the following file:

```
1: This is a sample file
2: used to show dynamic line numbers.
3:*See what happens when you use
4: Delete and Insert
5: (the D and I commands)
6: to edit text
7: in your file.
```

To delete a range of lines beginning with the current line, type

```
,6D
```

The result is as follows:

```
1: This is a sample file
2: used to show dynamic line numbers.
3:*in your file.
```

Note that the lines are automatically renumbered.

Name:

Edit

Syntax:

[<line>]

Function:

Edits line of text.

Comments:

When a line number is typed, EDLIN displays the line number and text; then, on the line below, EDLIN reprints the line number. The line is now ready for editing. You may use any of the EDLIN editing commands to edit the line. The existing text of the line serves as the template until RETURN is pressed.

If no line number is typed (that is, if only RETURN is pressed), the line after the current line (marked with an asterisk (*)) is edited. If no changes to the current line are needed and the cursor is at the beginning or end of the line, press RETURN to accept the line as is.

NOTE

> If RETURN is pressed while the cursor is in the middle of the line, the remainder of the line is deleted.

For example, assume that the following file exists and is ready to edit.

```
1: This is a sample file.
2: used to show
3: the editing of line
4:*four.
```

To edit line 4, type

```
4
```

The contents of the line are displayed with a cursor below the line.

```
4:* four.
4:*_
```

Now, using the COPYALL special editing key, type

```
INSERT number        4: number_
COPYALL  RETURN      4: number four.
                     5:*_
```

Name:

End

Syntax:

E

Function:

Ends the editing session.

Comments:

This command saves the edited file on diskette or hard disk, renames the original input file <filename> .BAK, and then exits EDLIN. If the file was created during the editing session, no .BAK file is created.

The E command takes no options. Therefore, you cannot tell EDLIN on which drive to save the file. The drive you want to save the file on must be selected when the editing session is started. If the drive is not selected when EDLIN is started, the file will be saved on the default drive. It will still be possible to copy the file to a different drive using the COPY command.

NOTE

> You must be sure that the storage device contains enough free space for the entire file. If this device does not contain enough free space, the write will be aborted and the edited file lost, although part of the file might be written out.

After execution of the E command, the MS-DOS system prompt is displayed.

Name:

Insert

Syntax:

```
[<line>]I
```

Function:

Inserts text immediately before the specified <line>.

Comments:

If you are creating a new file, you must give the I command before text can be typed (inserted). Text begins with line number 1. Successive line numbers appear automatically each time RETURN is pressed.

EDLIN remains in Insert mode until CTRL-C is typed. When the insert is completed and Insert mode has been exited, the line immediately following the inserted lines becomes the current line. All line numbers following the inserted section are incremented by the number of lines inserted.

If <line> is not specified, the default will be the current line number and the lines will be inserted immediately before the current line. If <line> is any number larger than the last line number, or if a pound sign (#) is specified as <line>, the inserted lines will be appended to the end of the file. In this case, the last line inserted will become the current line.

For example, assume that the following file exists and is ready to edit.

```
1: This is a sample file
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
6: to edit text
7:*in your file.
```

To insert text before a specific line that is not the current line, type <line>I. If the line is 7, the display appears as follows:

```
7I
```

The result is

```
7:_
```

Now, type the new text for line 7:.

```
7: and renumber lines
```

Then to end the insertion, press CTRL-Z on the next line.

```
8: CTRL-Z
```

Now type L to list the file. The result is as follows:

```
1: This is a sample file
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
6: to edit text
7: and renumber lines
8:*in your file.
```

To insert lines immediately before the current line, type

```
I
```

The result is

```
8:_
```

Now, insert the following text and terminate with a CTRL-Z on the next line:

```
8: so they are consecutive
9: CTRL-Z
```

Now to list the file and see the result, type L.

The result is as follows:

```
 1: This is a sample file
 2: used to show dynamic line numbers.
 3: See what happens when you use
 4: Delete and Insert
 5: (the D and I commands)
 6: to edit text
 7: and renumber lines
 8: so they are consecutive
 9:*in your file.
```

To append new lines to the end of the file, type

```
 10I
```

This produces the following:

```
10:_
```

Now, type the following new lines:

```
10: The insert command can place new lines
11: in the file; there's no problem
12: because the line numbers are dynamic;
13: they'll go all the way to 65533.
```

End the insertion by pressing CTRL-Z on line 14. The new lines will appear at the end of all previous lines in the file. Now type the list command, L.

The result is as follows:

```
 1: This is a sample file
 2: used to show dynamic line numbers.
 3: See what happens when you use
 4: Delete and Insert
 5: (the D and I commands)
 6: to edit text
 7: and renumber lines
 8: so they are consecutive
 9: in your file.
10: The insert command can place new lines
11: in the file; there's no problem
12: because the line numbers are dynamic;
13: they'll go all the way to 65533.
```

Name:

List

Syntax:

[<line>][,<line>]L

Function:

Lists a range of lines, including the two lines specified.

Comments:

Default values are provided if either one or both of the options are omitted. If you omit the first option, as in

,<line>L

the display will start 11 lines before the current line and end with the specified line. The beginning comma is required to indicate the omitted first option.

NOTE

If the specified <line> is more than 11 lines before the current line, the display will be the same as if you omitted both options.

If you omit the second option, as in

<line>L

23 lines will be displayed, starting with the specified <line>.

If you omit both parameters, as in

L

23 lines will be displayed — the 11 lines before the current line, the current line, and the 11 lines after the current line. If there are fewer than 11 lines before the current line, more than 11 lines after the current line will be displayed to make a total of 23 lines.

For example, assume that the following file exists and is ready to be edited.

```
 1: This is a sample file
 2: used to show dynamic line numbers.
 3: See what happens when you use
 4: Delete and Insert
 5: (the D and I commands)
  .
  .
  .
15:*The current line contains an asterisk.
  .
  .
  .
26: to edit text
27: in your file.
```

To list a range of lines without reference to the current line, type <line>,<line>L.

```
2,5L
```

The result is as follows:

```
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
```

To list a range of lines beginning with the current line, type, <line>L.

```
,26L
```

The result is as follows:

```
15:*The current line contains an asterisk.
  .
  .
  .
26: to edit text
```

To list a range of 23 lines centered around the current line, type only L.

```
L
```

The result is as follows:

```
 4:Delete and Insert
 5:(the D and I commands)
  .
  .
  .
13:The current line is listed in the middle of the range.
14:The current line remains unchanged by the L command.
15:*The current line contains an asterisk.
  .
  .
  .
26:to edit text.
```

Name:

Move

Syntax:

```
[<line>],[<line>],<line>M
```

Function:

Moves a range of text to the line specified.

Comments:

Use the Move command to move a block of text (from the first <line> specified to the second <line> specified) to another location in the file. The lines are renumbered according to the direction of the move. For example,

```
,+25,100M
```

moves the text from the current line plus 25 lines to line 100. If the line numbers overlap, EDLIN will display an "Entry error" message.

To move lines 20-30 to line 100, type

```
20,30,100M
```

Name:

Page

Syntax:

[<line>][,<line>]P

Function:

Pages through a file 23 lines at a time.

Comments:

If the first <line> is omitted, that number will default to the current line plus one. If the second <line> is omitted, 23 lines will be listed. The new current line becomes the last line displayed and is marked with an asterisk.

Name:

Quit

Syntax:

```
Q
```

Function:

Quits the editing session, does not save any editing changes, and exits to MS-DOS.

Comments:

EDLIN prompts you to make sure you don't want to save the changes.

Type Y if you want to quit the editing session. No editing changes are saved and no .BAK file is created. Refer to the End command in this chapter for information about the .BAK file.

Type N or any other character if you want to continue the editing session.

NOTE

When started, EDLIN erases any previous copy of the file with an extension of .BAK to make room to save the new copy. If you reply Y to the "Abort edit (Y/N)?" message, your previous backup copy will no longer exist.

For example,

```
Q
Abort edit (Y/N)?Y RETURN
```

ends the editing session and returns the system prompt:

```
A>_
```

Name:

Replace

Syntax:

[<line>][,<line>][?]R<string1> CTRL-Z <string2>

Function:

Replaces all occurrences of a string of text in the specified range with a different string of text or blanks.

Comments:

As each occurrence of <string1> is found, it is replaced by <string2>. Each line in which a replacement occurs will be displayed. If a line contains two or more replacements of <string1> with <string2>, then the line will be displayed once for each occurrence. When all occurrences of <string1> in the specified range are replaced by <string2>, the R command terminates and the asterisk prompt reappears.

If a second string is to be given as a replacement, then <string1> must be separated from <string2> with a CTRL-Z. <String2> must also be ended with a CTRL-Z RETURN combination or with a simple RETURN.

If <string1> is omitted, then Replace will take the old <string1> as its value. If there is no old <string1>, that is, this is the first replacement done, then the replacement process will be terminated immediately. If <string2> is omitted, then <string1> may be ended with a RETURN. If the first <line> is omitted in the range argument (as in ,<line>) then the first <line> will default to the line after the current line. If the second <line> is omitted (as in <line> or <line>,), the second <line> will default to #. Therefore, this is the same as <line>,#. Remember that # indicates the line after the last line of the file.

If <string1> is ended with a CTRL-Z and there is no <string2>, <string2> will be taken as an empty string and will become the new replace string. For example,

R<string2> CTRL-Z RETURN

will delete occurrences of <string1>, but

R<string1> RETURN and
R RETURN

will replace <string1> by the old <string2> and the old <string1> with the old <string2>, respectively. Note that "old" here refers to a previous string specified either in a Search or a Replace command.

If the question mark (?) option is given, the Replace command will stop at each line with a string that matches <string1>, display the line with <string2> in place, and then display the prompt "O.K.?". If you press Y or RETURN, then <string2> will replace <string1>, and the next occurrence of <string1> will be found. Again, the "O.K.?" prompt will be displayed. This process will continue until the end of the range or until the end of the file. After the last occurrence of <string1> is found, EDLIN displays the asterisk prompt.

If you press any key besides Y or RETURN after the "O.K.?" prompt, the <string1> will be left as it was in the line, and Replace will go to the next occurrence of <string1>. If <string1> occurs more than once in a line, each occurrence of <string1> will be replaced individually, and the "O.K.?" prompt will be displayed after each replacement. In this way, only the desired <string1> will be replaced, and you can prevent unwanted substitutions.

For example, assume that the following file exists and is ready for editing.

```
 1: This is a sample file
 2: used to show dynamic line numbers.
 3: See what happens when you use
 4: Delete and Insert
 5: (the D and I commands)
 6: to edit text
 7: in your file.
 8: The insert command can place new lines
 9: in the file; there's no problem
10: because the line numbers are dynamic;
11: they'll go all the way to 65533.
```

To replace all occurrences of <string1> with <string2> in a specified range, type

```
2,12 Rand CTRL-Z or RETURN
```

The result is as follows:

```
4: Delete or Insert
5: (the D or I commors)
8: The Insert commor can place new lines
```

Note that in the above replacement, some unwanted substitutions have occurred. To avoid these and to confirm each replacement, the same original file can be used with a slightly different command.

In the next example, to replace only certain occurrences of the first <string> with the second <string> , type

```
2? Rand CTRL-Z or RETURN
```

The result is as follows:

```
4: Delete or Insert
O.K.? Y
5: (The D or I commands)
O.K.? Y
5: (The D or I commors)
O.K.? N
8: The insert commors can place new lines
O.K.? N
*_
```

Now, type the List command (L) to see the result of all these changes.

```
  .
  .
4: Delete or Insert
5: (The D or I commands)
  .
8: The insert command can place new lines
  .
  .
```

Name:

Search

Syntax:

[<line>][,<line>][?]S<string>RETURN

Function:

Searches the specified range of lines for a specified string of text.

Comments:

The <string> must be ended with a RETURN. The first line that matches <string> is displayed and becomes the current line. If the question mark option is not specified, the Search command will terminate when a match is found. If no line contains a match for <string> , the message "Not found" will be displayed.

If the question mark option (?) is included in the command, EDLIN will display the first line with a matching string; it will then prompt you with the message "O.K.?". If you press either the Y or RETURN key, the line will become the current line and the search will terminate. If you press any other key, the search will continue until another match is found, or until all lines have been searched (and the "Not found" message is displayed).

If the first <line> is omitted (as in , <line> S<string>), the first <line> will default to the line after the current line. If the second line is omitted (as in <line> S<string> or <line>, S<string>), the second <line> will default to # (line after last line of file), which is the same as <line>,# S<string>. If <string> is omitted, Search will take the old string if there is one. (Note that "old" here refers to a string specified in a previous Search or Replace command.) If there is not an old string (i.e., no previous search or replace has been done), the command will terminate immediately.

For example, assume that the following file exists and is ready for editing.

```
 1: This is a sample file
 2: used to show dynamic line numbers.
 3: See what happens when you use
 4: Delete and Insert
 5: (the D and I commands)
 6: to edit text
 7: in your file.
 8: The Insert command can place new lines
 9: in the file; there's no problem
10: because the line numbers are dynamic;
11: *they'll go all the way to 65533.
```

To search for the first occurrence of the string "and", type

```
2,12 Sand RETURN
```

The following line is displayed:

```
4: Delete and Insert
```

To get the "and" in line 5, modify the search command by typing

```
SKIP1   COPYALL   ,12 Sand RETURN
```

The search then continues from the line after the current line (line 4), since no first line was given. The result is

```
5: (the D and I commands)
```

To search through several occurrences of a string until the correct string is found, type

```
1,? Sand
```

The result is

```
4: Delete and Insert
O.K.?_
```

If you press any key (except Y or RETURN), the search continues, so type N here.

```
O.K.? N
```

Continue.

```
5: (the D and I commands)
O.K.?_
```

Now press Y to terminate the search.

```
O.K.? Y
*_
```

To search for string XYZ without the verification (O.K.?), type

```
SXYZ
```

EDLIN will report a match and will continue to search for the same string when you issue the S command.

```
S
```

EDLIN reports another match:

```
S
```

EDLIN reports the string is not found.

Note that string defaults to any string specified by a previous Replace or Search command.

Name:

Transfer

Syntax:

[<line>]T<filename>

Function:

Inserts (merges) the contents of <filename> into the file currently being edited at <line>. If <line> is omitted, then the current line will be used.

Comments:

This command is useful if you want to put the contents of a file into another file or into the text you are typing. The transferred text is inserted at the line number specified by <line> and the lines are renumbered.

Name:

Write

Syntax:

```
[<n>]W
```

Function:

Writes a specified number of lines to disk from the lines that are being edited in memory. Lines are written to disk beginning with line number 1.

Comments:

This command is meaningful only if the file you are editing is too large to fit into memory. When you start EDLIN, it reads lines into memory until memory is 3/4 full.

To edit the remainder of your file, you must write edited lines in memory to disk. Then you can load additional unedited lines from disk into memory by using the Append command.

NOTE

If you do not specify the number of lines, lines will be written until memory is 3/4 full. No action will be taken if available memory is already more than 3/4 full. All lines are renumbered, so that the first remaining line becomes line number 1.

ERROR MESSAGES

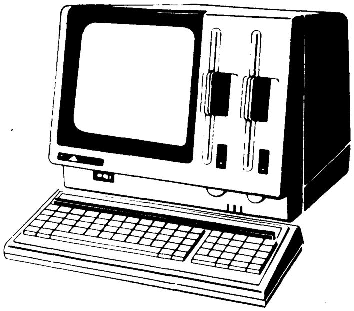
When EDLIN finds an error, one of the error messages in Table 5-2 is displayed.

Table 5-2 EDLIN Error Messages

ERROR MESSAGE	CORRECTIVE PROCEDURE
Cannot edit .BAK file — rename file	You attempted to edit a file with a filename extension of .BAK. .BAK files cannot be edited because this extension is reserved for backup copies. If you need the .BAK file for editing purposes, you must either RENAME the file with a different extension; or COPY the .BAK file and give it a different filename extension.
No room in directory for file	When you attempted to create a new file, either the file directory was full or you specified an illegal disk drive or an illegal filename. Check the command line that started EDLIN for illegal filename and illegal drive entries. If the command is no longer on the screen and if you have not yet typed a new command, the EDLIN start command can be recovered by pressing PF3. If this command line contains no illegal entries, run the CHKDSK program for the specified drive. If the status report shows that the disk directory is full, remove the disk. Insert and format a new disk.
Entry Error	The last command typed contained a syntax error. Retype the command with the correct syntax and press RETURN.
Invalid Parameter	You specified a switch other than /B when starting EDLIN. Specify the /B switch when you start EDLIN.
Insufficient memory	There is not enough memory to run EDLIN. You must free some memory by writing files to disk or by deleting files before restarting EDLIN.

Table 5-2 EDLIN Error Messages (cont'd)

File not found	The filename specified during a Transfer command was not found. Specify a valid filename when issuing a Transfer command.
Must specify destination number	A destination line number was not specified for a Copy or Move command. Reissue the command with a destination line number.
Not enough room to merge the entire file	There was not enough room in memory to hold the file during a Transfer command. You must free some memory by writing some files to disk or by deleting some files before you can transfer this file.



Appendix A

Files Supplied on the MS-DOS Diskette

Table A-1 lists the program and data files supplied on the MS-DOS diskette in alphabetical order.

Table A-1 Files Supplied on the MS-DOS Diskette

FILES	DESCRIPTION
AUTOEXEC.BAT	Batch file containing the PATH command to the INSTALL directory file (see below) for the initial installation of MS-DOS.
AUXCG.CHR	Auxiliary character data set file. Stores the special character set you create for display on the screen. This file is used with CHR.COM (see below).
BLINK.COM	Utility that adjusts the blink rate for the cursor.
CAT.COM	Directory catalog utility. Displays the names and sizes of all the files in a directory in alphabetical order.
CHKDSK.COM	Diskette check utility. Verifies the contents of a diskette.
CHR.COM	Auxiliary character generator utility. Creates special character sets for display.
CIPHER.COM	Cipher utility. Encrypts files to restrict unauthorized access.
COMMAND.COM	The command processor.
CONFIG.SYS	System configuration file. Keeps track of required system information, such as the default input and output devices and the number of storage device I/O buffers.

Table A-1 Files Supplied on the MS-DOS Diskette (Cont.)

FILES	DESCRIPTION
CREF.EXE	Cross-reference utility (MS-CREF). Creates a cross-reference of symbols in an assembly language source file. Used with the Macro Assembler.
DEBUG.COM	Debugging utility (DEBUG). Corrects errors in assembled or compiled object files.
DISKCOPY.COM	Disk copy utility. Copies the contents of an entire diskette to another.
DISP.COM	Auxiliary character display utility. Displays the currently loaded auxiliary character set.
DISP1.COM	Default character display utility. Displays the APC default character set.
DOCU <DIR>	Directory file containing assembly source listings for the I/O System routines. Also supplies the CONFIG.DOC file that explains the use of CONFIG.SYS, if used.
EDLIN.COM	Line editor utility (EDLIN). Used to create and modify data files (text and language source code).
FC.EXE	File comparison utility. Compares two data (text or source) or object files, showing similarities.
FIND.EXE	Character string search utility. Locates character strings typed at the keyboard in one or more files.
FORMAT.COM	Diskette formatting utility. Analyzes and prepares diskettes for storing program and data files.
HDFORMAT.EXE	Hard disk formatting utility. Prepares a hard disk for the storage of files.
INSTALL <DIR>	Directory file containing batch files that perform the initial installation for MS-DOS. These files are not needed again, so they are not copied to working copies of the MS-DOS diskette when created by the user.

Table A-1 Files Supplied on the MS-DOS Diskette (Cont.)

FILES	DESCRIPTION
IO.SYS	MS-DOS system program that controls peripheral devices. This is a "hidden" file. It does not appear on a directory display when the DIR command is used.
KEY.COM	Function key assignment utility. Assigns specific functions to the dual-mode (PF) keys on the APC keyboard.
LIB.EXE	Library management utility. Creates library files to hold the object modules generated by language compilers and the Macro Assembler, and allows for updating and managing the member files.
LINK.EXE	Linker utility. Links individual object modules or whole libraries to create executable .COM and .EXE files.
LOCATE.EXE	Executable file converter. Converts EXE. type files to .COM files.
MASM.EXE	The Macro Assembler (MACRO-86). Translates 8086 or 8088 assembly language code into the hexadecimal machine code format constituting an assembled program.
MORE.COM	Screen display utility. Routes output to the screen, allowing for its review.
MSDOS.SYS	MS-DOS system program that performs major control functions in the system. This is a "hidden" file. It does not appear on a directory display when the DIR command is used.
MUSICTST.COM	Music generator program.
POW.COM	Automatic power-off utility. Shuts off the APC by software control.
PRINT.COM	Print spooling utility. Performs the background printing operation.

Table A-1 Files Supplied on the MS-DOS Diskette (Cont.)

FILES	DESCRIPTION
RDCPM.COM	CP/M TO MS-DOS diskette copy utility. Copies a file from a single-density diskette formatted under C/PM to the same kind of storage device formatted under MS-DOS.
RECOVER.COM	Diskette file recovery utility. Restores access to files on a diskette when there are bad sectors.
SETCOM.COM	Data communications interface utility. Readies the APC for data communications by defining the operating parameters of the RS-232-C serial interface unit.
SORT.EXE	Filter utility. Processes the input data, sorting it based on a designated field and sending it to the screen or files in a rearranged format.
SYS.COM	MS-DOS software program transfer utility. Copies MSDOS.SYS and IO.SYS from the MS-DOS diskette to another formatted diskette. Normally used to make back-up copies of these two programs.
TIMEROFF.COM	Timer-off utility. Turns off the automatic update and display functions from the APC clock/ calendar. The current time display is restored by executing TIMERON.COM.
TIMERON.COM	Timer-on utility. Turns the APC clock/calendar function back on, displaying the correct time.
VISUAL.COM	Demonstrates the various display attributes for screen characters.

Advanced Personal Computer™

NEC
NEC Information Systems, Inc.

USER'S COMMENTS FORM

Document: MS-DOS System User's Guide

Document No.: 819-000101-6001 Rev. 00

Please suggest improvements to this manual.

Please list any errors in this manual. Specify by page.

From:
Name ______
Title ______
Company ______
Address ______
Dealer Name ______
Date: ______

Please cut along this line.

Seal or tape all edges for mailing-do not use staples.

FOLD HERE

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO.105 BOXBOROUGH, MA

POSTAGE WILL BE PAID BY ADDRESSEE

NEC Information Systems, Inc.
Dept: Publications — APC
1414 Massachusetts Avenue
Boxborough, MA 01719

FOLD HERE

Seal or tape all edges for mailing-do not use staples.